

BBN Report No. 3438

JP
NW

ADA035278

SPEECH UNDERSTANDING SYSTEMS

Final Report

November 1974 - October 1976

Volume V: Trip

Sponsored by
Advanced Research Projects Agency
ARPA Order No. 2904

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

DDC
RECEIVED
FEB 8 1977
D

JP

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by ONR under Contract No. N00014-75-C-0533.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Advanced Research Projects Agency or the U.S. Government.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER BBN Report No. - 3438 - Vol-5	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER Technical progress rpt.
4. TITLE (and Subtitle) SPEECH UNDERSTANDING SYSTEMS. Volume I. Final Technical Progress Report 30 October 1974 - 29 October 1976 Trip.		5. TYPE OF REPORT & PERIOD COVERED Final Tech. Prog. Report 30 Oct. 1974 - 29 Oct. 1976
7. AUTHOR(s) Woods, M. Bates, G. Brown, B. Bruce, C. Cook J. Klovstad, J. Makhoul, B. Nash-Webber, R. Schwartz, J. Wolf, V. Zue.		6. PERFORMING ORG. REPORT NUMBER BBN Report No. 3438
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. ✓ 50 Moulton Street Cambridge, Ma. 02138		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0533, ✓ 15 WARRPA Order - 2904
11. CONTROLLING OFFICE NAME AND ADDRESS ONR Department of the Navy Arlington, Va. 22217		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 5D30
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE Dec 1976
		13. NUMBER OF PAGES 115 123p.
		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Commerce for sale to the general public.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Acoustic-phonetic experiment facility, acoustic-phonetic recognition, acoustic-phonetic rules, artificial intelligence, ATN grammars, audio- response generation, budget management, computational linguistics, control strategies, data base, dictionary expansion, discourse model, fact retrieval, formal command language, formant tracking, grammars, HWIM, knowledge sources,		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This five-volume final report is a review of the BBN speech understanding system, covering the last two years of the project from October 1974 through October 1976. The BBN speech understanding project is an effort to develop a continuous speech understanding system which uses syntactic, semantic, and pragmatic support from higher level linguistic knowledge sources to compensate for the inherent acoustic indeterminacies in continuous spoken utterances. These knowledge sources are integrated with		

DC FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

060 100

542

19. Key words (cont'd.)

lexical retrieval, likelihood ratio, linear prediction, multi-component systems, natural language retrieval system, natural language understanding, parametric modeling, parsing, pattern recognition, phonetic labeling, phonetic segmentation, phonological rules, phonology, pragmatic grammar, pragmatics, probabilistic labeling, probabilistic lexical retrieval, prosodics, question-answering, recognition strategies, resource allocation, response generation, scoring philosophy, semantic interpretation, semantic networks, semantics, shortfall algorithm, shortfall density, signal processing, spectral matching, speech, speech analysis, speech generation, speech synthesis, speech recognition, speech understanding, SUR, syntax, synthesis-by-rule, system organization, task model, user model, word verification.

20. Abstract (cont'd.)

sophisticated signal processing and acoustic-phonetic analysis of the input signal, to produce a total system for understanding continuous speech. The system contains components for signal analysis, acoustic parameter extraction, acoustic-phonetic analysis of the signal, phonological expansion of the lexicon, lexical matching and retrieval, syntactic analysis and prediction, and inferential fact retrieval and question answering, as well as synthesized text or spoken output. Those aspects of the system covered in each volume are:

- | | |
|-------------|--|
| Volume I. | Introduction and Overview |
| Volume II. | Acoustic Front End |
| Volume III. | Lexicon, Lexical Retrieval and Control |
| Volume IV. | Syntax and Semantics |
| Volume V. | The Travel Budget Manager's Assistant |

White Section	<input checked="" type="checkbox"/>
Buff Section	<input type="checkbox"/>
UNCLASSIFIED	<input type="checkbox"/>
CLASSIFICATION	
EXEMPTION/AVAILABILITY CODES	
AVAIL. ORG. OR SPECIAL	
A	

SPEECH UNDERSTANDING SYSTEMS

Final Report

November 1974 - October 1976

ARPA Order No. 2904

Program Code No. 5D30

Name of Contractor:
Bolt Beranek and Newman Inc.

Effective Date of Contract:
30 October 1974

Contract Expiration Date:
29 October 1976

Amount of Contract: \$1,966,927

Contract No. N00014-75-C-0533

Principal Investigator:
William A. Woods
(617) 491-1850 x361

Scientific Officer:
Marvin Denicoff

Title:
SPEECH UNDERSTANDING SYSTEMS

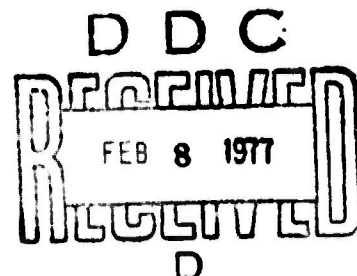
Editor:
Bonnie Nash-Webber
(617) 491-1850 x227

Sponsored by
Advanced Research Projects Agency
ARPA Order No. 2904

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by ONR under Contract No. N00014-75-C-0533.

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited



The BBN Speech Understanding System: Final Report

This is one of five volumes of the BBN Speech Understanding System Final Report. The Table of Contents for the entire set is given below.

Volume I. Introduction and Overview

- A. Introduction
- B. Design Philosophy of HWIM
- C. Overview of final system
- D. Design of final performance test and performance analysis overview
- E. Future
- F. References
- G. Appendices
 - 1. Sample set of sentence types
 - 2. Sample trace of an utterance being processed
 - 3. Publications
 - 4. Comprehensive Index to Technical Notes

Volume II. Acoustic Front End

- A. Acoustic Front End
- B. Acoustic-Phonetic Recognition
- C. A Speech Synthesis-by-Rule Program
- D. Verification
- E. References
- F. Appendices
 - 1. Dictionary Phonemes
 - 2. List of APR labels
 - 3. List of APR rules
 - 4. Parameters for Scoring

Volume III. Lexicon, Lexical Retrieval and Control

- A. Dictionary
- B. Phonological rules
- C. Dictionary Expansion
- D. Lexical Retrieval
- E. Control Strategy
- F. Performance
- G. References
- H. Appendices
 - 1. Annotated phonological rules
 - 2. Format and examples of dictionary files
 - 3. Result summaries for each token (TRAVELDICT and BIGDICT)
 - 4. Performance Results for Strategy Variations
 - 5. BIGDICT and TRAVELDICT listings
 - 6. Dictionary Expansion - A User's Guide

Volume IV. Syntax and Semantics

- A. Parsers
- B. Grammars
- C. Prosodics
- D. References
- E. Appendices
 - 1. Listing of MIDGRAM Grammar
 - 2. Sample Parse-Interpretations
 - 3. Parser trace

Volume V. TRIP

- A. Introduction
- B. The Travel Budget Manager's Assistant
- C. Flow of Control
- D. Linguistic Processing
- E. Execution
- F. Response Generation
- G. Conclusions
- H. References
- I. Appendices
 - 1. Data Base Structures
 - 2. Example Parses and Interpretations
 - 3. Methods
 - 4. Generation Frames
 - 5. A Generated Description of the Stored Trips

Acknowledgements

To the following people who contributed to the development of the BBN Speech Understanding System:

William A. Woods, Principal Investigator
Madeleine Bates
Geoffrey Brown
Bertram C. Bruce
Laura Gould
Craig C. Cook
Gregory Harris
Dennis H. Klatt
John W. Klovstad
John I. Makhoul
Bonnie L. Nash-Webber
Richard M. Schwartz
Jared J. Wolf
Victor W. Zue

To our secretaries - Beverly Tobiason, Kathleen Starr and Angela Beckwith - for the exceptional diligence, competence, and good humor they have shown throughout the assembly of this report.

"Had we but world enough and time..."

Andrew Marvell, To His Coy Mistress

TABLE OF CONTENTS

	Page
A. Introduction	1
A.1 Natural Communication with a Computer;	1
A.2 Types of Knowledge Needed;	1
A.3 Representations and Processing;	3
B. The Travel Budget Manager's Assistant	7
B.1 Travel Budget Management;	7
B.2 An Example	9
C. Flow of Control;	15
D. Linguistic Processing	19
D.1 Design Philosophy	19
D.2 Parsing;	20
D.2.1 SPEECHIFY	20
D.2.2 Pragmatic Grammar	20
D.3 Semantic Interpretation	22
D.3.1 General Method	22
D.3.2 Special Cases	24
E. Execution	28
E.1 SEMNET - The Network Utility Package;	28
E.1.1 Network Components	28
E.1.2 Implementation	29
E.2 Command Language;	30
E.3 Optimization	31
E.4 Data Base Structures;	33
E.4.1 Characterization	33
E.4.2 Implementation	37
E.5 Inference	39
E.6 Discourse Model;	40
F. Response Generation	47
F.1 Overview	47
F.2 Text Response Generation; and	47
F.3 Speech Synthesis	55
G. Conclusion	58
H. References	59
I. Appendices	61
I.1 Data Base Structures	61
I.2 Example Parses and Interpretations	75
I.3 Methods	91
I.4 Generation Frames	101
I.5 A Generated Description of the Stored Trips	113

List of Figures

1. Higher level knowledge used in speech understanding
2. Representation of higher level knowledge
3. Layers of knowledge in TRAVELNET
4. Example sentences
5. Information given the user upon entering a dialogue with the system
6. Incremental simulations of the travel budget manager's assistant
7. Structure of the IBM assistant
8. Flow of control during speech understanding
9. Flow of control during text processing
10. A small portion of the pragmatic grammar
11. Examples of PARSER/TRIP interactions
12. Example parse structures for time expressions
13. A time point structure
14. A contract and a budget
15. A budget item
16. A trip
17. Factual knowledge in TRAVELNET (numbers are approximate number of nodes for each type)
18. Fictitious links
19. Information used in estimating the cost of a trip
20. Trace of a cost computation showing use of METHODS
21. A discourse state
22. A registration fee

A. Introduction

A.1 Natural Communication with a Computer

Volumes 1 through 4 of this final report explore in some detail the workings of HWIM, the BBN speech understanding system. To some extent this volume also covers aspects of speech understanding per se, but it does so in the larger context of communication between a person and a computer. Volume 5 is concerned with the result of speech understanding as well as the process of achieving it. Questions such as:

(1) What constitutes the understanding of an utterance?

and

(2) What should the system do once it has understood?

become paramount here. We begin to see HWIM as one part of a system whose goal is to provide assistance to a person, and to make that assistance available in the most natural possible way, via both spoken and written natural language.

The person who speaks to HWIM is assumed to be a travel budget manager. HWIM is thus the speech understanding part of a travel budget manager's assistant. This volume is primarily a discussion of the travel budget manager's (TBM) assistant with an emphasis on the ways in which the system facilitates natural communication between person and computer. For example,

- (1) The state of the discourse and the data base are made available to HWIM while processing an utterance to constrain its possible interpretations, thus increasing the likelihood of successful understanding (Section D.2.2).
- (2) Inference mechanisms make possible looser expression of questions and commands by the manager, by divorcing semantic interpretations from explicit data structures (Section E.5).
- (3) English-like responses are used both to give explicitly requested information, and to exhibit the program's state of knowledge (or ignorance) regarding the on-going dialogue (Section F.2).

A.2 Types of Knowledge Needed

In order to carry out its role effectively, any computer-based assistant must have a large amount of knowledge in a readily accessible form. This knowledge is needed for understanding utterances, carrying out commands, answering questions, and generating responses. For various computational, practical, conceptual and historical reasons, such knowledge in HWIM is expressed in different representation languages: augmented

transition networks, semantic networks, arrays and list structures. This section is primarily a representation-independent discussion of the kinds of knowledge used by the assistant. Because acoustic, phonetic or phonological knowledge have been covered elsewhere (Volumes 1-3), we will be focusing on so-called "higher-level knowledge."

What follows is a brief survey of the major types of knowledge needed by the TBM assistant. Those types required in speech understanding per se are shown also in Figure 1.

(1) Basic maintenance knowledge - Knowledge concerning the representation and use of other knowledge. This is the kind of knowledge that would be needed in even a "blank" system; i.e., one that had no word- or domain-specific knowledge.

(2) Computational knowledge - Knowledge about data base structures and allowable inferencing procedures on them. Much of this knowledge exists as METHODS associated with link names in the semantic network (see Section E.5).

(3) Lexical knowledge - Knowledge of words, their inflections, parts of speech and phonemic spellings.

(4) Syntactic knowledge - Knowledge of grammatical structures and structurally-conveyed meaning.

(5) Semantic knowledge - Knowledge of how words are related and used. This is primarily used in constructing responses (see Section F).

(6) World knowledge - Knowledge of a specific "real" world, e.g. that people (and not computer assistants) take trips.

(7) Task knowledge - Knowledge of the task domain, i.e. that a manager is concerned with maintaining an accurate record of all trips, taken or planned, and not going over his budget.

(8) User knowledge - Knowledge about each travel budget manager at BBN, what group s/he belongs to, and what s/he may know about the data base.

(9) Factual knowledge - Knowledge of specific projects, trips, budgets and conferences. It is distinguished from world knowledge in that it

changes rapidly as events occur. In fact, whereas WIM's world knowledge is essentially fixed, its factual knowledge changes every time a trip is taken or money shifted from one budget item to another (see Section E.4).

(10) Discourse knowledge - Knowledge of idealized discourse, i.e. what type of utterance is likely to be produced in a given state of the discourse, and knowledge of the current discourse state, e.g., the current topic, the objects available for anaphoric reference, the speaker and the current date. The latter knowledge is used to relax constraints in the pragmatic grammar (see Section D.2.2). For example, if a conference is under discussion, then "What is the registration fee?" is a meaningful utterance. If not, then the speaker would be required to say something like, "What is the registration fee for the XXX conference?" (see Section E.6).

LEXICAL - "ENTERED" IS THE PAST FORM OF "TO ENTER".

SYNTACTIC - A SENTENCE MAY START WITH A COMMAND VERB, E.G.,
"SCHEDULE..."

SEMANTIC - THE BENEFACTIVE CASE FOR "SCHEDULE" SHOULD BE
FILLED BY A PERSON,

WORLD - BUDGET ITEMS HAVE ALLOCATIONS AND COVER TRIPS.

TASK - IN THE TRAVEL BUDGET MANAGEMENT DOMAIN, PEOPLE
CAN SCHEDULE OR TAKE TRIPS.

USER - ONLY CERTAIN USERS CAN MODIFY BUDGET TOTALS.

FACTUAL - WOODS' FIRST NAME IS "BILL".

DISCOURSE - "WHEN IS THAT MEETING?" IS OK IN CERTAIN CONTEXTS.

FIGURE 1. HIGHER LEVEL KNOWLEDGE USED IN SPEECH UNDERSTANDING

A.3 Representations and Processing

Deciding what knowledge is needed by the TBM assistant is only the first step in designing such a system. One must also choose a

representation for it which can be applied to various tasks, such as constraining the possible interpretations of speech input or answering questions. The history of HWIM (see [Nash-Webber and Bruce, 1976]) and the IBM assistant is filled with examples of changes in the representation of knowledge.

One interesting example starts with a consideration of semantic interpretation. In one version of the system syntactic knowledge was packaged in an augmented transition network (ATN) and semantic knowledge in both a semantic network (Section E.1) and LUNAR-style [Woods, Kaplan, and Nash-Webber, 1972] semantic interpretation rules. The close interaction between syntactic and semantic knowledge sources that was needed for speech understanding, plus the task oriented nature of the language allowed the travel budget manager, led to a merging of much of this knowledge in our pragmatic grammar (Section D.2.2). With the added semantic and pragmatic knowledge interpretations could be produced as easily as parse trees. In the final version of the system, nearly executable interpretations are produced directly by the parser using the pragmatic grammar.

The semantic network still exists, however, as a repository of semantic information used for tasks other than interpretation (e.g. optimization of interpretations, Section E.3; execution of interpretation, Section E.5). In fact, we now have a division of higher level knowledge into two major data structures, the ATN and the semantic network. The current division (illustrated in Figure 2) reflects experience with many different processing demands.

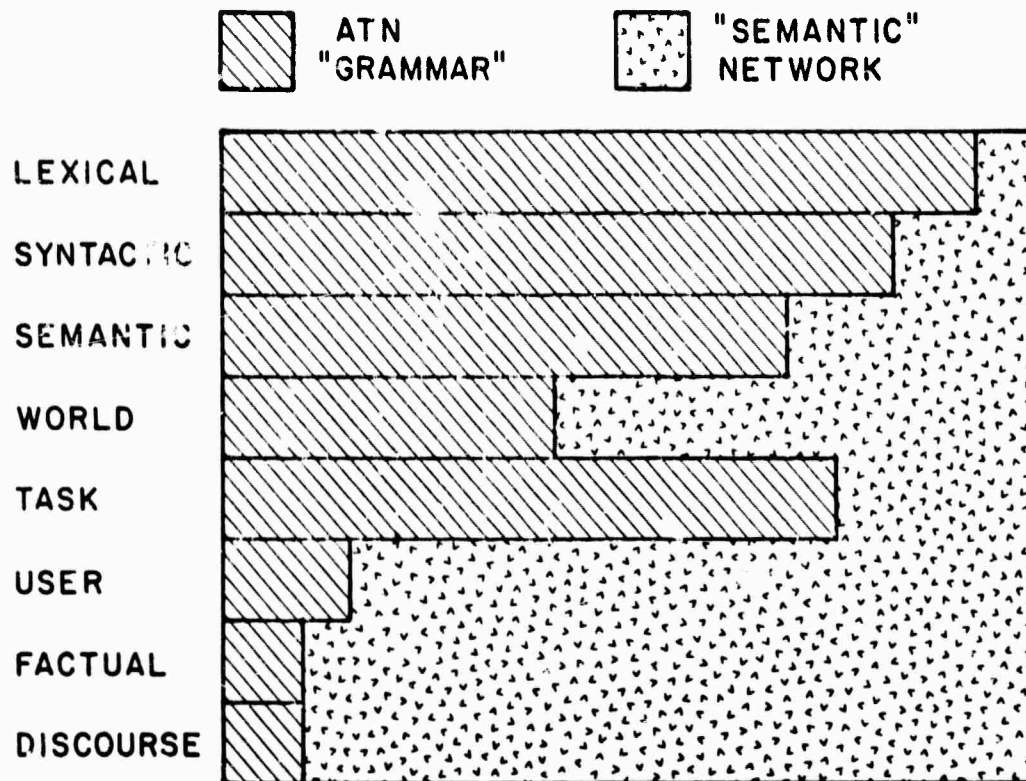


FIGURE 2. REPRESENTATION OF HIGHER LEVEL KNOWLEDGE

The general rule that emerged was that the more fluid types of knowledge (e.g. data about a specific trip) fit best in the semantic network where changes could be made easily. Thus, over half of the semantic network (see Figure 3) is devoted to the representation of factual knowledge.

<u>TYPE</u>	<u>% OF TRAVELNET</u>
NET MAINTENANCE (INCLUDING 280 LINK NAMES)	13
COMPUTATIONAL	2
LEXICAL	1
SYNTACTIC	9
SEMANTIC	6
WORLD	7
TASK	2
USER	1
FACTUAL	58
DISCOURSE	1

FIGURE 3. LAYERS OF KNOWLEDGE IN TRAVELNET

Knowledge that was relatively more fixed (e.g. syntactic structures) fit best in the ATN. The division was feasible at all only because of the possibility of close communication between the processors operating on these data structures.

Our experience with the ATN, semantic network and semantic interpretation rules was repeated many times within these data structures. Later sections of this report discuss the choices made for various types of knowledge that the TBM assistant uses.

B. The Travel Budget Manager's AssistantB.1 Travel Budget Management

The task of the system is to be an assistant to a travel budget manager, helping to record the trips taken or proposed and to produce such summary information as the total money allocated. The task is a simplified example of many other resource management problems as well and is an initial step toward developing an intelligent automatic assistant. Figure 4 shows some examples of the kinds of sentences the system may be called upon to understand.

Lyn is going to Chicago in August.

How much money is left in the budget?

Estimate the cost for a trip to L.A. for 5 days.

Print out all taken trips.

What is the fare between Boston and Chicago?

What trips are scheduled for Laura to California?

When is Chip's trip to Pittsburgh?

Add Bill to the Pittsburgh trip.

How long is that trip?

Enter a trip for Jerry to Austin.

The account is 11349.

He is leaving on March fifteenth.

I estimate the cost at three hundred and fifty dollars.

Change the account to 11273.

FIGURE 4. EXAMPLE SENTENCES

The intended scope of the system can be seen in Figure 5 in the information typed to the user upon entering the system.

You are now talking to the travel budget manager's assistant. The assistant can help you keep track of money in the budget and whether that money has been spent or committed. It also keeps a record of each trip, including the traveler, the starting place, the destination, the dates and/or length of the trip, the account to be charged, the estimated and actual costs, and whether the trip is part of the actual costs, and whether the trip is part of the actual budget or a hypothetical budget. Budgets include a record of trips that have been taken, scheduled, or merely proposed. There is also built-in knowledge of fares, dates and times, locations, and other information needed to maintain the budget. The assistant does not make policy decisions nor does it plan trips.

The assistant accepts commands to modify its data base and answers questions about the information it has stored. Statements are interpreted as commands to add, remove, or change information. In trying to perform some command, the system may need to ask you a question. When it does, its strongest expectation is for you to give a direct answer to the question, but you may say that you don't know or simply go on to further commands or questions. You may also have the system prompt you for the information needed to complete the description of a trip, e.g., "What else do you need to know?"

FIGURE 5. INFORMATION GIVEN THE USER UPON
ENTERING A DIALOGUE WITH THE SYSTEM

Early in the speech project we ran simulations in order to develop and circumscribe the TBM assistant (see Figure 6). In the simulations, one person sitting at terminal A played the role of the travel budget manager, typing in sentences as if he were talking to a complete travel budget manager's assistant. Another person, at terminal B, intercepted his sentences, translating them into the formal command language of the system [see Section E.2], and passing the translations to the retrieval component for execution. These simulations also provided a source for dialog protocols and new words to include in the dictionary.

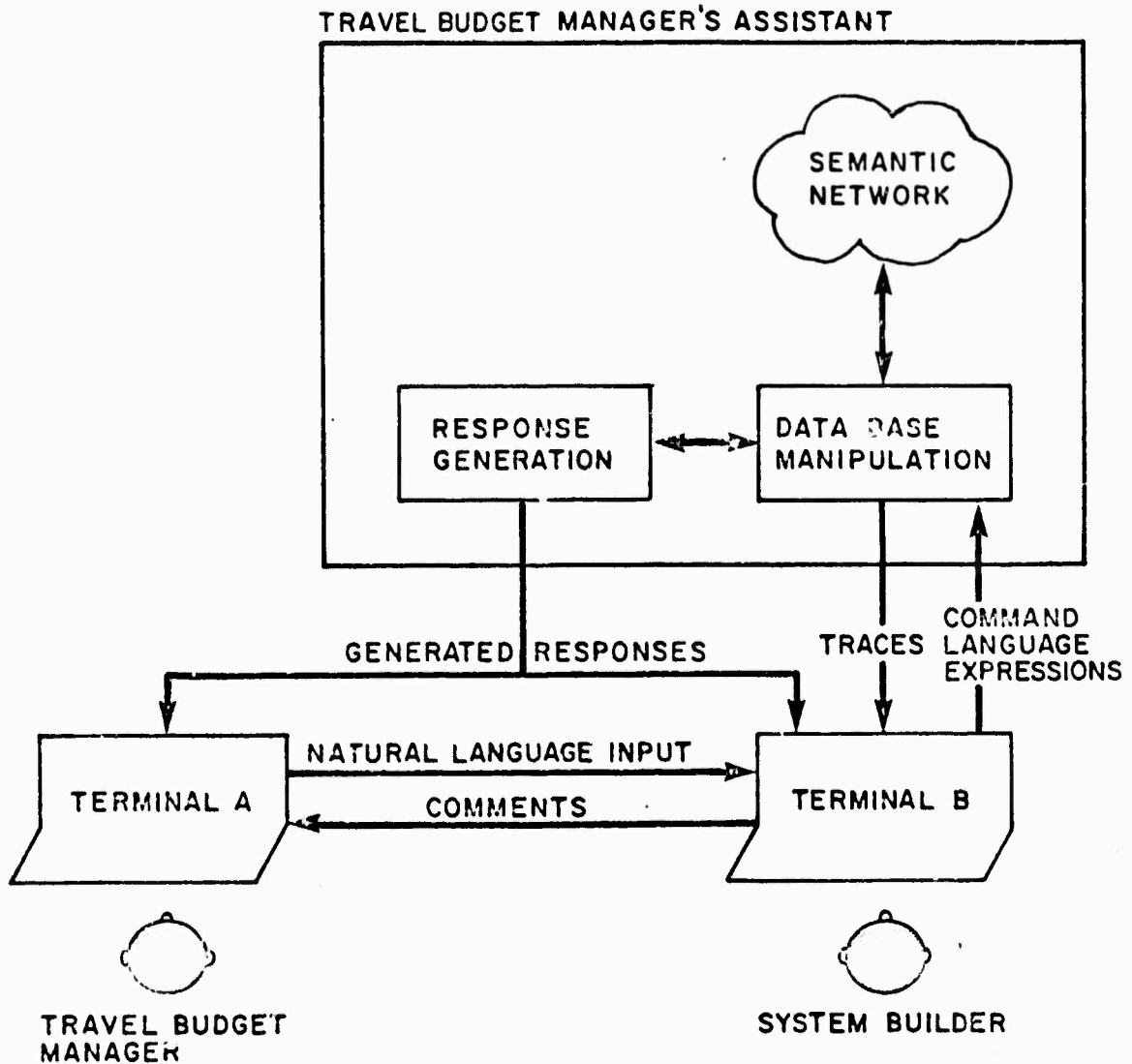


FIGURE 6. INCREMENTAL SIMULATIONS OF THE TRAVEL BUDGET MANAGER'S ASSISTANT

B.2 An Example

In this section we give a partial trace of the TBM assistant's processing of two typed-in sentences. Since the difference between typed and correctly understood spoken input is only marginal by the time it reaches the TBM assistant, we will discuss things here only in terms of typed input.

In general, the sentences shown here would occur as part of an ongoing dialogue. While their interpretations and resulting responses might be significantly different in context, what can be seen here is the flow of control within the system, the types of interpretations produced, the inference capabilities, and response generation.

We begin with the travel budget manager (i.e., the human) typing:

when did Bill go to Mexico?

The program makes a first pass on the input converting the words as typed into words as they appear in its dictionary and removing punctuation. For example, "\$23.16" would become "twenty three dollar-s and sixteen cent-s" (See Section D.2.1). In this case the modified word list is simply

(WHEN DID BILL GO TO MEXICO)

which is what we hope would be the word list of a spanning theory, had the sentence been uttered to HWIM. From here on processing typed input is indistinguishable from processing spoken input.

The parser in HWIM uses a pragmatic grammar (see Section D.2.2; Volume 4, Section B) which contains significant static knowledge of the travel budget management domain. This task specific grammar makes possible simultaneous parsing and semantic interpretation (see Section D.3; Volume 4, Section B). This is in contrast with the two phase method used in the LUNAR parser [Woods, Kaplan and Nash-Webber, 1972] and earlier versions of HWIM parser which produced purely syntactic parse trees that were subsequently given to a semantic interpreter to transform into executable interpretations.

For the example sentence, the parser used to produce the form:

```
S Q
  QADV WHEN
  SUBJ NPR BILL
  AUX TNS PAST
      VOICE ACTIVE
  VP V GO
      PP PREP TO
          NP NPR MEXICO
  ADV THEN
```

Now, taking advantage of the task specific nature of the grammar, we get in addition:

```
[FOR: THE A0009 / (FINDQ: LOCATION (COUNTRY MEXICO))
  T ; (FOR: THE A0008 / (FINDQ: PERSON (FIRSTNAME BILL))
    : T ; (FOR: ALL A0010 /
      (FINDQ: DB/TRIP (TRAVELER A0008)
        (DESTINATION A0009)
        (TIME (BEFORE NOW)))
    : T ; (OUTPUT: (GET: A0010 TIME)]
```

The interpretation is a quantified expression which can (almost) be directly executed. Translated it says, roughly:

Find the location which has the country name "Mexico" and call it A0009. Find the person whose first name is "Bill" and call him A0008. Find in turn, all trips whose traveler is A0008, whose destination is A0009, and which occurred prior to now. As each is enumerated, label it A0010 and output its specific time.

The T's which appear in the interpretation are there in place of potential restrictions which might have been applied to the location, person, or trip classes being searched.

Before the interpretation can be executed it undergoes an optimization (see Section E.3). In this example interpretation, several things need to be changed. First, there is no entity called "LOCATION" in the semantic network data base which serves the purpose implied in the interpretation. There are cities, states, coasts, countries, etc. and there is a link called "LOCATION", but no concept with instances as we might expect. The interpretation thus references a fictitious node. The optimizer recognizes this and invokes a special finding function (FINDLOC) corresponding to the fictitious node. Second, the FINDQ: function in the interpretations is a convenience which gets replaced by the more general function FIND: (see Section E.2). Finally, the optimizer corrects for some minor incompatibilities due to a changeover in interpretation method (e.g.,

(GET: A0010 TIME) -> (GET: A0010 (QUOTE TIME))

and

(TIME (BEFORE NOW)) -> (TIME (QUOTE PAST)))

The final optimized interpretation is:

```
[FOR: THE A0009 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
                          (COUNTRY (QUOTE MEXICO)))
  : T ; (FOR: THE A0008 / (FIND: (INSTANCE/OF (QUOTE PERSON))
                              (FIRSTNAME (QUOTE BILL)))
    : T ; (FOR: ALL A0010 /
          (FIND: (INSTANCE/OF QUOTE DB/TRIP))
            (TRAVELER A0008)
            (DESTINATION A0009)
            (TIME (QUOTE PAST)))
    : T ; (OUTPUT: (GET: A0010
                      (QUOTE TIME])
```

The optimized interpretation is placed at the rear end of a queue of to-be-executed commands. The queue represents a first step towards a "demand model of discourse" (see Section E.6). In principle it can contain previous queries or commands from the speaker as well as system initiated commands. In the existing version of the system, however, most utterances translate into single commands which are directly executed.

The (FOR: --) expression above is a LISP form which can be evaluated directly. The FOR: function manages quantification in the expected way (see Section E.2). Here it looks for a unique location and a unique person which match the respective descriptions. There is a single place whose country name is "Mexico", i.e. the country, Mexico. However, there are 8 items in the data base representing people whose first name is "Bill". Given the apparent inconsistency between the data base and the command, FOR: is forced to take the initiative of the dialogue and ask the manager for help. The response generation program (Section F) produces the question:

Do you mean BILLY DISKIN, BILL HUGGINS, BILL LEVISON, BILL PATRICK, BILL PLUMMER, BILL RUSSELL, BILL MERRIAM, OR BILL WOODS?

If the AUDIOFILE flag is set, the response generation program also produces a list of phonemes and prosodic markers that is given to the speech synthesis program (Section F.3 and Volume 2, Section D) which produces a waveform file that can be played out as an audio response. In this case the input produced for the synthesis program is the following:

```
(D UW !2 # Y UW !2 # M IY !2 N # B IH !2 * L IY !0 # D IH !2 * S
K IX !- N # , # B IH !2 L # HH AH !2 * G IX !- N Z # , # B IH !2
L # L EH !2 * V IX !- * S AX !- N # , # B IH !2 L # P AE !2 * DX
AXR !- K # , # B IH !2 L # P L AH !2 * M AXR !- # , # B IH !2 L #
R AH !2 * S AX !- L # , # B IH !2 L # M EH !2 * R IY !0 * AX !- M
# , # AO !2 R # B IH !2 L # W UH !2 D Z ?)
```

When the travel budget manager's assistant asks a question (as in this example) it expects an answer. This does not mean that non-answers will be misunderstood, but only that otherwise "incomplete" utterances may be accepted. Here the parser will allow a name as a complete utterance. In this context, the travel budget manager types his/her second sentence:

Bill Woods.

As did the first sentence, this sentence undergoes a pre-pass to produce the word list

(BILL WOODS)

which the parser interprets as

(! THE A0011 PERSON ((FIRSTNAME BILL) (LASTNAME WOODS)))

The optimized interpretation is:

(IOTA 1 A0011 / (FIND: (INSTANCE/OF (QUOTE PERSON))
 {FIRSTNAME (QUOTE BILL))
 (LASTNAME (QUOTE WOODS))} : T)

This interpretation uses the IOTA operator (see Section E.2), a function that returns the 1 item in the class defined by the (FIND: - -) expression which meets the restriction, T (i.e. no restriction). In this case there is exactly one item matching the description, namely BILL WOODS. Having now a unique DESTINATION and a unique TRAVELER, FOR: can find all trips (i.e. DB/TRIP) whose TIME is PAST and for each, output its TIME.

Finding the trips, checking TRAVELER, DESTINATION, and TIME can require considerable search in the data base. For example, the DESTINATION of a DB/TRIP is computed from the DESTINATIONS of the LEG/OF/TRIPS making it up. The DESTINATION of a LEG/OF/TRIP may have to be computed from the PURPOSE of the LEG/OF/TRIP, e.g. attending a specific conference. Information about trips, budgets, conferences, etc. is never assumed to be complete since it is not so in the real world. Furthermore, the range of askable questions precludes computing in advance all the implicit

information. The METHODS that do these computations are discussed in Section E.5.

In this example there is only one trip that matches the description and its time is printed out (or spoken) by the response generation programs (Section F):

BILL WOODS' TRIP from BOSTON MASSACHUSETTS to JUAREZ MEXICO was from October 15th, 1975 to October 17th.

(B IH !2 L # W UH !2 D Z # ' # T R IH !2 P # FW # F R AH !2 M # B
AO !2 # S T AX !- N # M AE !? # S AX !- # CH UY !2 # S IX !- T S
FW # T UW !2 # W AA !1 # R EH !2 Z # M EH !2 K # S IX !- # K OW
!1 # W AH !2 Z # FW # F R AH !2 M # AX !- K # T OW !2 # B AXR !-
F IH !1 F # T IY !2 N TH # , # N AY !2 N # T IY !1 N # S EH !2
V AX !- N # T IY !0 # F AY !2 V # FW # T UW !2 # AX !- K # T OW
!2 # B AXR !- # S EH !1 # V AX !- N # T IY !2 N TH %.)

C. Flow of Control

There is an idealized view of natural language communication by computer that postulates a four stage process of parsing, interpretation, execution, and response generation. In stage 1, the system determines the syntactic structure of the input. In stage 2 it transforms the syntactic structure into a procedural representation of its meaning (perhaps just "Add this fact.") In stage 3 the interpretation is executed or performed. In stage 4 any response is formulated and given out.

Often stages 1 and 2 are combined, that is, the system accepts its input and produces an executable interpretation. Often stage 4 is reduced to "Print out the computed data" or "Insert the computed data into a response frame and print it out". In any case, these stages are familiar, and it is convenient to describe the operation of the TBM assistant in terms of them. Thus, Section D covers parsing and interpretation; Section E covers execution; and Section F covers response generation. This linear presentation should not be interpreted as a claim that the corresponding human language processes are neatly separable, nor that they are so even within the TBM assistant. We shall see many examples where parsing and interpretation become fused, where execution-level inferences are crucial to correct parsing and where response generation depends upon information derived during parsing or execution.

This section gives an overview of the system organization and a sketch of the flow of control during the processing of an utterance. Sections D-F then fill out the details.

A fanciful, and partial, representation of the TBM assistant, including HWIM, is shown in Figure 7. In the figure, ovals indicate TENEX "forks" (runnable processes) and clouds indicate major data structures. Dashed lines represent inter-fork communication channels and solid lines represent data structure accessibility. Most of this volume is concerned with programs within the fork labeled TRIP. Missing from Figure 7 are two major data structures, the pragmatic grammar (Volume 4) and the segment lattice (Volume 2), as well as an indication of the actual flow of control during processing. The discussion to follow elaborates the figure by describing various configurations used in the actual running of the system.)

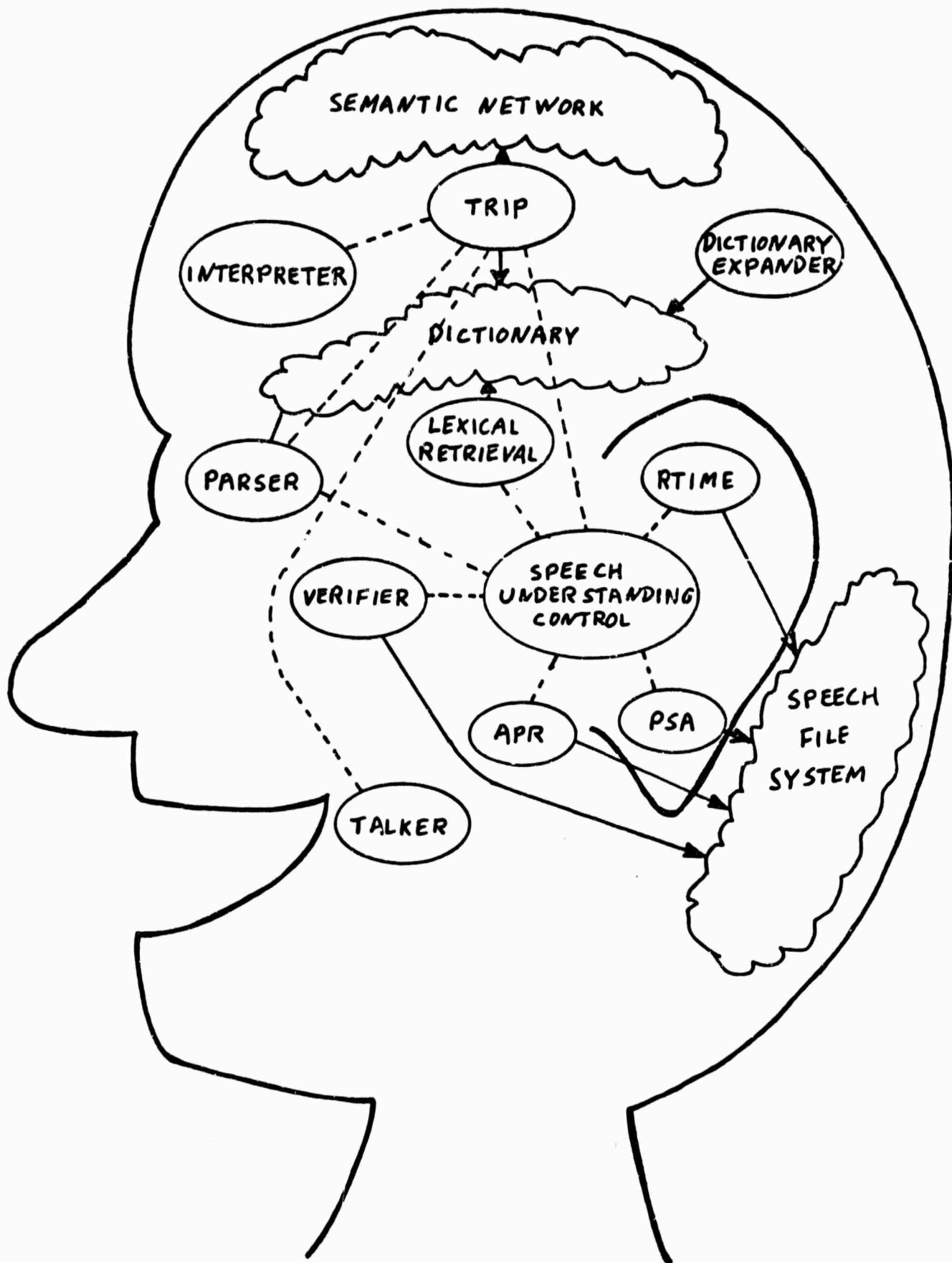


FIGURE 7. STRUCTURE OF THE TBM ASSISTANT

When HWIM is being tested, the top level fork is usually the one labeled SPEECH UNDERSTANDING CONTROL. After signal acquisition (RTIME), signal processing (PSA), and acoustic phonetic recognition (APR), the running system reduces to five forks as shown in Figure 8.

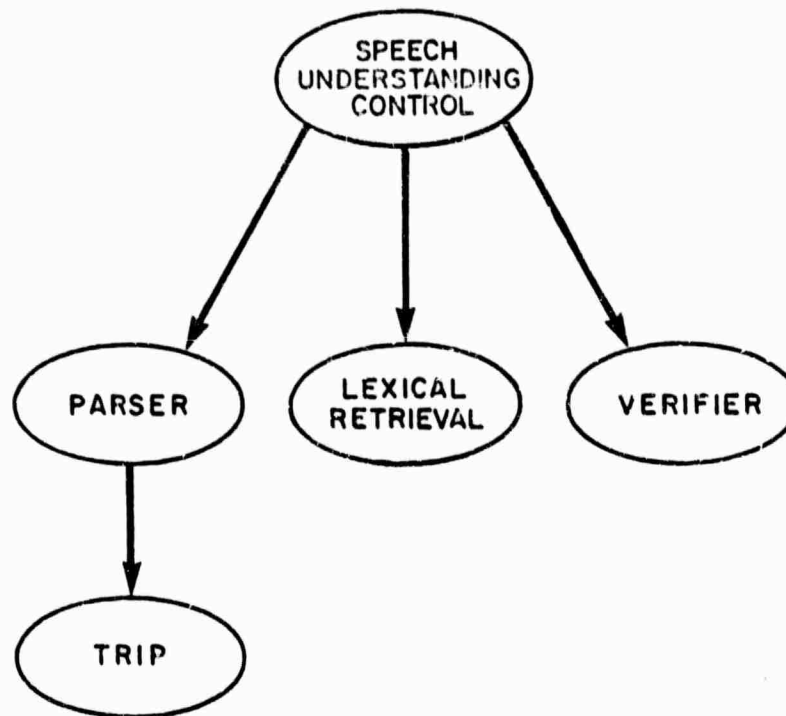


FIGURE 8. FLOW OF CONTROL DURING SPEECH UNDERSTANDING

In that configuration, the syntactic component (PARSER) functions as a linguistic consultant for HWIM, proposing new words to be matched in the segment lattice and confirming "theories". Here PARSER calls TRIP as a subprocess to verify tests on arcs in the grammar. Another configuration has TRIP as the top fork, with SPEECH UNDERSTANDING CONTROL as its subprocess. This is the configuration which would be used in an actual dialogue with a travel budget manager, since TRIP contains the world and discourse knowledge necessary to engaging in the dialogue. (See Section D.3 for a discussion of the INTERPRETER fork).

When the TBM assistant engages in a text dialogue, the system reduces to just two processes, TRIP and PARSER. In that mode PARSER is a subprocess to TRIP, though it may itself invoke TRIP as its own subprocess

to do the above mentioned verification. To see one possible flow of control, (Figure 9), consider the sentence -

Enter a trip for Jerry Wolf to New York.

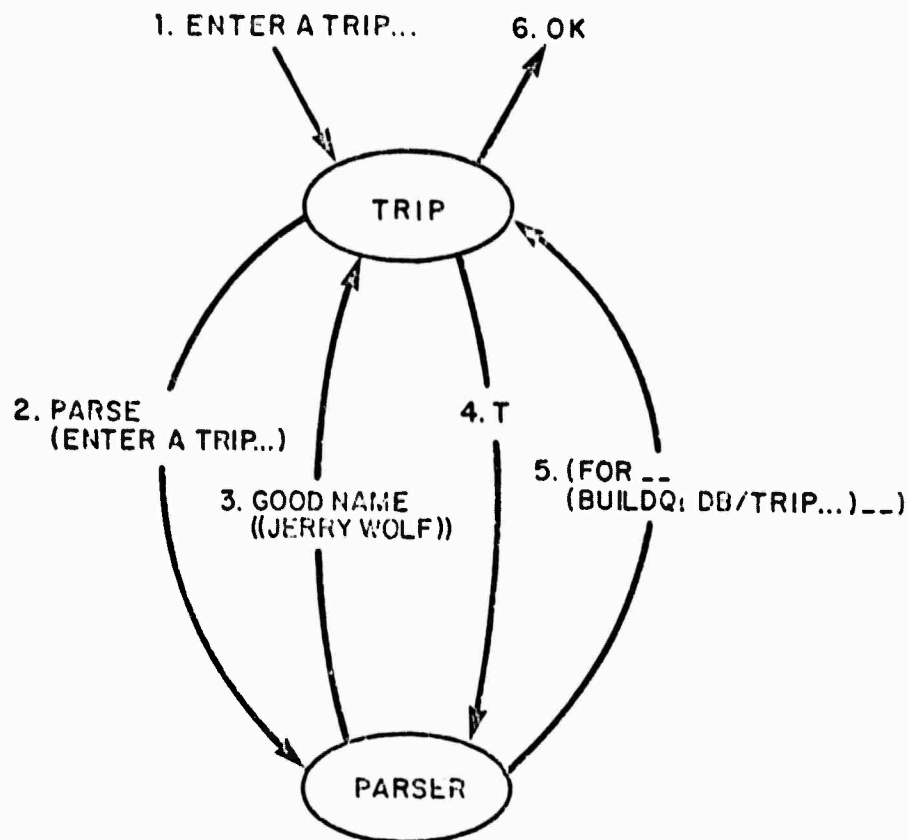


FIGURE 9. FLOW OF CONTROL DURING TEXT PROCESSING

TRIP reads the input and applies SPEECHIFY (see Section D.2.1). It then calls PARSER to parse and interpret the sentence. During processing, PARSER may encounter a test, e.g. "Does the name 'Jerry Wolf' denote a known person?", which is to be performed by TRIP. Control passes to TRIP, and then back to PARSER again. When the parse is complete, control returns to TRIP. Ambiguity in the input can cause a question to be formulated for the travel budget manager, which subsequently causes a return to PARSER, and so on to the end of the session.

D. Linguistic Processing

D.1 Design Philosophy

The process of understanding written natural language requires the integration of diverse knowledge sources. An optimal process must apply various types of knowledge in a balance that avoids over- or under-constraining the set of potential accounts of the input. This balance is even more crucial for spoken input.

Ultimately the understanding process should produce a meaning representation for the natural language input, whether the input be written or spoken. This representation may be produced directly or via some number of intermediate structures (cf. "contingent knowledge structures" [Bobrow and Brown, 1975]). The exact form of these intermediate structures varies from one system to another, but typically includes such things as parse trees.

In the TBM assistant, integration of knowledge sources for linguistic processing is accomplished by means of (1) an ATN grammar that merges much of the syntactic, semantic and pragmatic knowledge the system has, (2) a semantic network that represents remaining linguistic and world knowledge, and (3) tests on arcs in the grammar to effectively link the two representations. These structures were used to produce the account of the input that serves as a representation of its meaning.

This section gives but a sketch of the linguistic processing in the TBM assistant. First (in Section D.2.1) we consider the morphological analysis program (SPEECHIFY) that produces a dictionary-compatible version of the input. Next (in Section D.2.2) we briefly discuss the pragmatic grammar and its linkage to the semantic network. (A fuller discussion is given in Volume 4.) Following the pragmatic grammar we consider semantic interpretation (in Sections D.3.1 and D.3.2.) The final evolved version of the system has now these stages of linguistic processing:

- (1) (a) spoken input, or
 (b) written input
- (2) dictionary-compatible words
- (3) semantic interpretation, ready to be optimized and executed

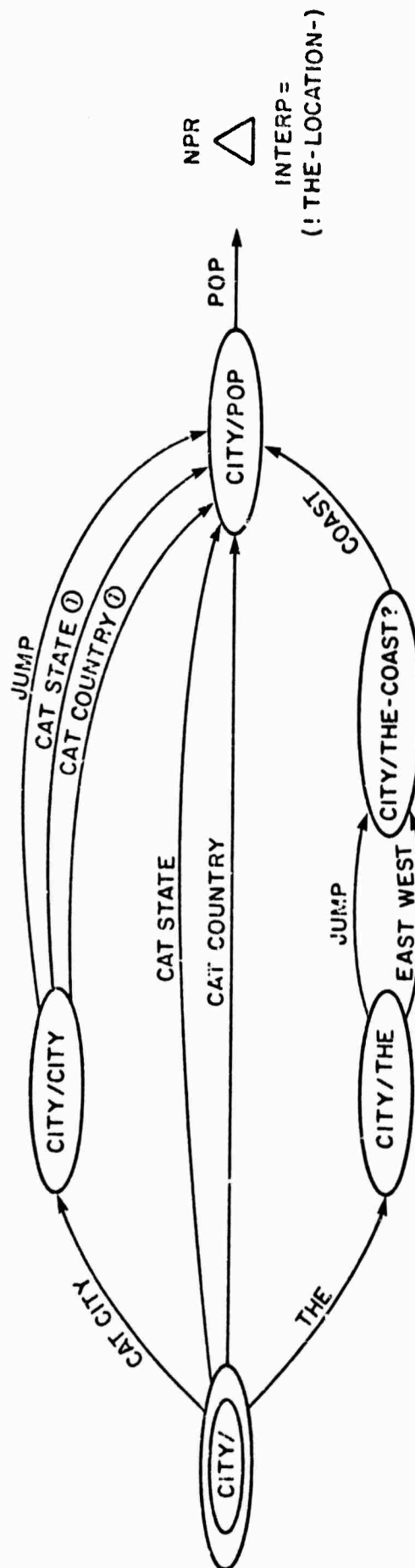
D.2 Parsing

D.2.1 SPEECHIFY

The function SPEECHIFY has an essential role in audio response generation (see Section F.3), and also in the text version of the TBM Assistant. SPEECHIFY performs three major types of conversions. The first is to break up inflected words. Using a slightly modified version of the algorithm in Winograd [1971], SPEECHIFY converts "budgeted" to "budget-ed", "trips" to "trip-s", "Wolf's" to "Wolf -s" (where "-s" is the dictionary entry for the possessive morpheme), and so on. In each case the converted forms contain only words in the expanded dictionary (Volume 3, Section C). The second type of conversion is to replace numbers (including cardinals, ordinals, digit strings, and monetary figures) with a corresponding word string. For example, "\$354.78" becomes "three hundred and fifty four dollar-s and seventy eight cent-s", "3547 miles" becomes "three thousand five hundred and forty seven mile-s", "trip number 3547" becomes "trip number thirty five forty seven", "October 8th, 1975" becomes "October eighth nineteen-m seventy five" (where "nineteen-m" is the dictionary entry for the modifier form of "nineteen," and "account 11273" becomes "account one one two seven three". Note that these conversions require context sensitive checks on the surrounding words since, for example, amounts of money and trip numbers are pronounced differently. The third conversion is idiosyncratic to the dictionary. For example, certain multi-word phrases are concatenated, e.g., "Los Angeles" becomes "Los@Angeles", and homographs with different pronunciations are distinguished, e.g., "estimated" becomes "estimate-v-ed" (where "estimate-v" is the dictionary entry for the verb and "estimate-n", the entry for the noun form).

D.2.2 Pragmatic Grammar

One of the major data structures used in the TBM assistant is the pragmatic grammar. (A small example from the grammar is shown in Figure 10). The grammar represents a significant amount of semantic, world and task knowledge as well as the syntactic knowledge usually associated with a grammar. A full discussion of it can be found in Volume 4.



① CALL TRIP FORK TO VERIFY THAT THIS MATCHES

FIGURE 10. A SMALL PORTION OF THE PRAGMATIC GRAMMAR

Despite the inclusion of non-syntactic knowledge in the grammar, it is not a complete knowledge source by itself for linguistic processing. It is, however, closely linked to the semantic network via tests on the arcs. These tests allow the grammar to capture more volatile constraints on the input, such as those provided by the discourse model (Section E.6) or the factual data base (Section E.4). Examples of interactions between the PARSER fork using the pragmatic grammar and the TRIP fork using the semantic network are shown in Figure 11.

1. Is "JERRY WOODS" A VALID NAME?
2. Is "SAN FRANCISCO CALIFORNIA" A VALID PLACE?
3. WHAT WORDS CAN GO WITH "SPEECH" AS A PROJECT DESCRIPTOR (E.G., "UNDERSTANDING")?
4. Is "WHAT IS THE REGISTRATION FEE?" ALLOWABLE IN THIS CONTEXT?
5. Is "HOW MUCH IS IN THE BUDGET?" ALLOWABLE IN THIS CONTEXT?
6. Is "WHEN IS THAT MEETING?" ALLOWABLE IN THIS CONTEXT?
7. Is "NOVEMBER 15TH," ALLOWABLE IN THIS CONTEXT?

FIGURE 11. EXAMPLES OF PARSER/TRIP INTERACTIONS

D.3 Semantic Interpretation

D.3.1 General Method

As mentioned in Section C, parsing and interpretation can be viewed as distinct stages in the understanding of an utterance. In fact, an early version of the system had these processes in separate TENEX forks (see Figure 7). In that configuration, TRIP called PARSER for a syntactic parse tree, and then INTERPRETER for a semantic interpretation based on that tree.

With the grammar encoding semantic and pragmatic, as well as syntactic information, it is possible for the grammar to build a procedural "meaning" representation as well as a purely syntactic one. The grammar builds interpretations by accumulating in registers the semantic head, quantifier, and links of the nodes being described in the sentence (see Volume 4, Section B for a more complete description). For example, the sentence

I will go to the ASA meeting.

yields an interpretation in the command language (Section E.2) of the form

```
(FOR: THE A0018 / (FINDQ: DB/MEETING
                    (SPONSOR ASA)) : T ;
 (FOR: 1 A0019 / (BUILDQ: DB/TRIP
                    (TO/ATTEND A0018)
                    (TRAVELER SPEAKER))
                 : T ; T))
```

This interpretation is built up in the following way. The PUSH arc that looks for a constituent describing a person at the start of the sentence will transform the pronoun "I" into the link-node pair (TRAVELER SPEAKER) and return this as the interpretation of that constituent. The word "will" adds (TIME (AFTER NOW)) to the list of link-node pairs being accumulated. (The grammar does not accept constructions like "will have gone," so "will" can always be interpreted as marking a future event.) The word "go" sets the semantic head to DB/TRIP. Next, the constituent "the ASA meeting" produces

```
(TO/ATTEND (! THE Y DB/MEETING ((SPONSOR ASA))))).
```

(The ! indicates that a FOR: expression will have to be built as part of the interpretation.) The top level of the grammar has thus accumulated the link-node pairs

```
(TIME (AFTER NOW))
(TRAVELER SPEAKER)
(TO/ATTEND (!--))
```

with the semantic head DB/TRIP. The appropriate action (in this case a BUILDQ:) is created, and the necessary quantificational expressions are expanded around it.

D.3.2 Special Cases

Even when PARSER and INTERPRETER existed as separate forks, there were a number of special cases which produced a hybrid system (partly "interpret after parsing" and partly "interpret while parsing"). These cases included names, places and time references, for which structures were built by PARSER and passed to TRIP unscathed by INTERPRETER. The reason for these special cases was analogous to the reason which later determined the change in the general interpretation method: the semantic knowledge necessary to build executable structures was available at parsing time. The final version of the TBM assistant retains some of these special cases. While a totally uniform interpretation method (with corresponding uniformity in FOR: expressions, etc.) has conceptual advantages, it has disadvantages in terms of efficiency and convenience.

For example, date and time expressions [Bates and Bruce, 1975] have a special form in the command language in order to eliminate the possibility of multiple embedded quantifiers which the optimization program could not process effectively. If each noun phrase in the time expression

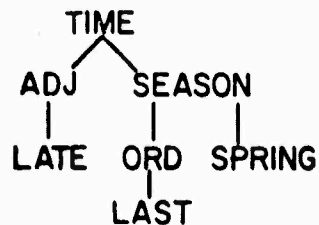
At 2:00 pm on the last Tuesday in March, 1975

were to result in its own quantified form, its interpretation might resemble the vastly unwieldy

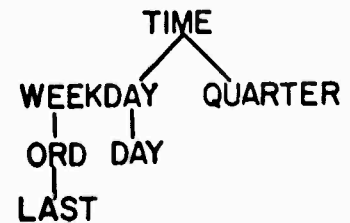
```
(FOR: THE A0001 / (FIND: (INSTANCE/OF (QUOTE YEAR))
                        (VALUE 1975))) : T ;
(FOR: THE A0002 / (FIND: (INSTANCE/OF (QUOTE MONTH))
                        (VALUE (QUOTE MARCH)))) : T ;
(FOR: LAST A0003 /
(FIND: (INSTANCE/OF (QUOTE WEEKDAY))
        (VALUE (QUOTE TUESDAY)))) : T ;
(FOR: THE A0004 /
(FIND: (INSTANCE/OF (QUOTE TIME/OF/DAY))
        (VALUE (QUOTE 2:00 PM))
```

Instead, the parser builds a parse structure which is not analyzed by the usual FOR: quantification program. Examples of these structures are shown in Figure 12.

LATE LAST SPRING, LATE IN LAST SPRING



THE LAST DAY OF THE QUARTER



EARLY IN THE FIRST QUARTER OF THE FISCAL YEAR

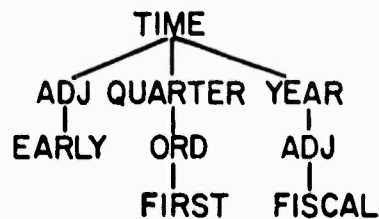
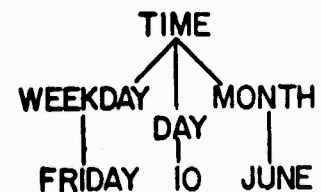
FRIDAY THE 10th OF JUNE

FIGURE 12. EXAMPLE PARSE STRUCTURES FOR TIME EXPRESSIONS

A special function, TIMEBUILD, takes the parsed time expression directly and builds the appropriate data base structures. For example, the phrase, "on a Tuesday in June, 1975" parses into

(TIME (WEEKDAY TUESDAY)(MONTH JUNE)(YEAR 1975))

which TIMEBUILD uses to build the data base structure shown in Figure 13. To do this, TIMEBUILD must consider the ordinals and adjectives and perform appropriate transformations on the values for each subunit.

Let us follow how TIMEBUILD processes the MONTH portion of a TIME structure. (Other portions of the TIME parse structure are processed in a similar way.) If there is no ORD (ordinal) link then the MONTH number for the month (e.g., August -> 8) is stored as is. If there is no month value, as in "this month", then the ORD link is used to calculate the appropriate MONTH and YEAR from the corresponding values on NOW, where NOW is a globally accessible time point which represents the current time. Note that for time expression we are treating "this," "next," and "last" as

ordinals. If there is both an ORD link and a month value, then the interpretation depends on the type of ordinal. For instance, "next" is interpreted as the first future occurrence of the specified month, e.g., if NOW is June, 1975, then "next August" means August, 1975 and "next May" means May, 1976.

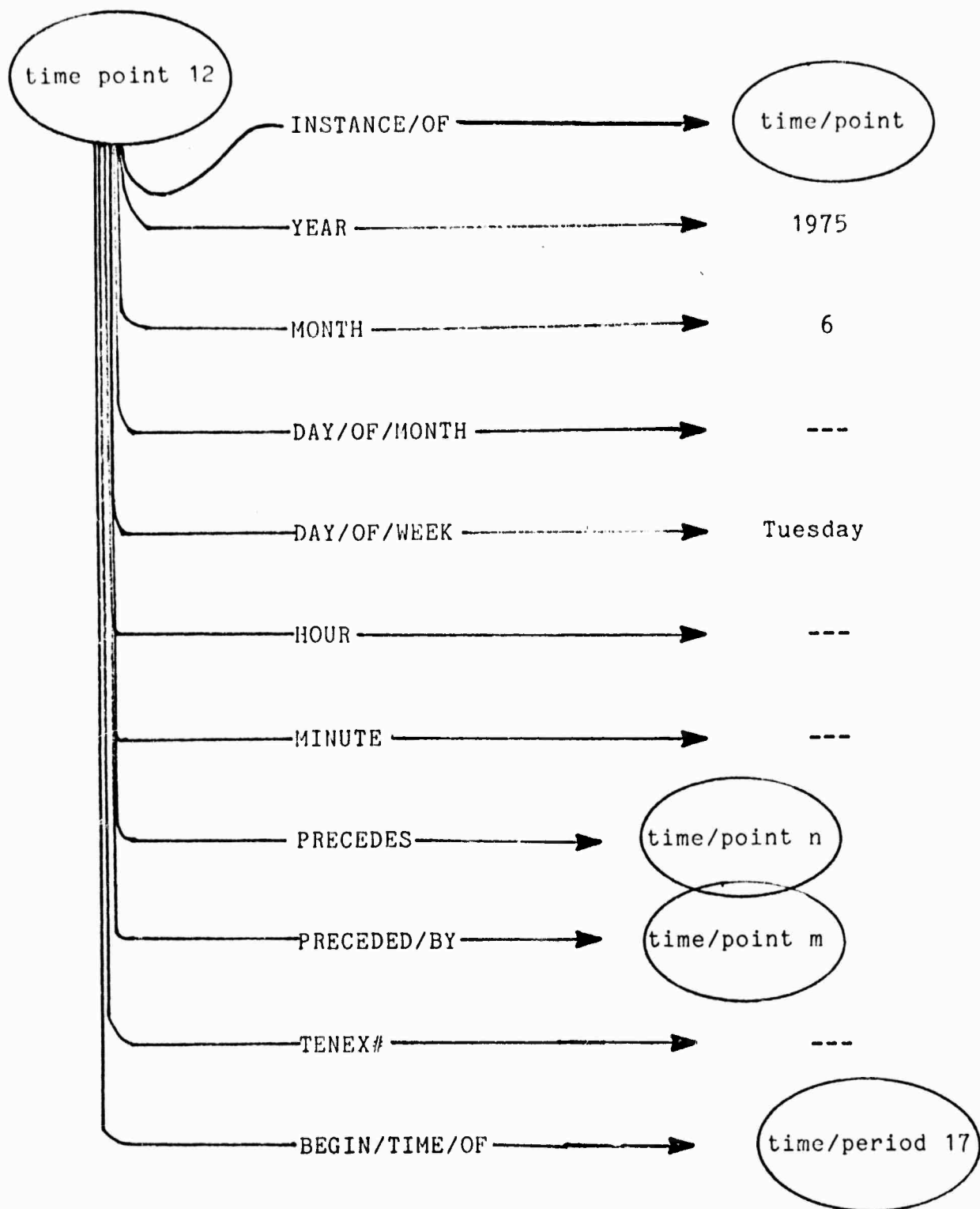


FIGURE 13. A TIME POINT STRUCTURE

E. Execution

E.1 SEMNET - The Network Utility Package

E.1.1 Network Components

Most of the TBM assistant's knowledge to be discussed in this volume is represented in the SEMNET formalism. Within the SEMNET formalism, there are two principal entities making up a semantic network: nodes and links. A node is a place at which information about a conceptual entity is collected and organized. A link is a directed association either between two nodes, or between a node and some information outside the network. A particular node->link->node triple is termed an arc.

Nodes may correspond to words, objects, events, etc. -- whatever one wants to have treated as a unique conceptual entity. A node may either be named, by associating with it a LISP print name, or be nameless. Independently, it may possess an "ego" which specifies the reason for its existence as a separate entity. For example, there could be one node whose name is "LYNN COSELL" and another whose ego is "LYNN COSELL as the TRAVELER of TRIP 7".

Nodes are connected to each other in this formalism via named links, called relations if they are two-way connections or properties if the connection is one-way. Properties may also be used to associate information outside the network with a node. For example, both names and egos are implemented as properties of a node, called PNAME and EGO respectively. The bi-directionality of relations is effected by means of link inverses. When a relation link of type R is established between nodes A and B, then a link of type R-inverse is automatically established between B and A.

All network links are named, and each linkname has its own associated node in the semantic net. While this may be treated as an invisible implementation decision by the network designer, one may also take advantage of it, as we have in the HWIM network, as a place to specify facts true of all arcs with a given linkname. For example, it can be used to store the name of the relation's inverse, such logical properties of a link as whether several arcs with that linkname entering a node should be treated as ANDed or ORed and how arcs of that type associate with other arcs. As a result, the distinction between "primitive" links and built-up

relations [Shapiro, 1971] is not made. For example, consider the network fragment:

639 - LOCATION

TYPE	LINKNAME
FINDFN	FINDLOC
*REL	(LOCATION/OF)
LINK/OF	(CITY) (DB/CONFERENCE) (DB/GROUP)
SYNTAX/CLASS	(CITY)
METHODS	1349
VALUE/CLASS	(CITY)
INSTANCE/OF	(LINKNAME)

1251

INSTANCE/OF	(DB/CONFERENCE)
DB/CREATOR	[CHIP BRUCE 276]
CREATE/TIME	1220
TIME	503
SPONSOR	(ASA)
ATTENDED/BY	594 698 1252
LOCATION	[STLOUIS MISSOURI 551]

Here LOCATION is both a conceptual entity (node 639) and the name of a relation (used in node 1251). Its existence as a conceptual entity (or node) allows us to specify explicitly such information as its possible values. LINK/OF is itself a link that indicates which data base structures use a particular link, e.g. LOCATION is used in the structures for CIT's, DB/CONFERENCEs and DB/GROUPs.

E.1.2 Implementation

The actual data structure in which the semantic network is stored in the current SEMNET formalism is an INTERLISP array, with each node corresponding to a single array element. A node is uniquely identified by its position in the array, e.g. item 1, item 2, etc., where this integer is called the node's SREF (for semantic referent). Each element of the array can hold two LISP pointers, one of which is used for the list of relational arcs leaving the node, the other for the list of properties. Both of these lists are stored in LISP property list format. Alternatively, there can be a pointer to a location on a file where these lists are stored. The latter method is used until the nodes are first accessed [Nash-Webber, 1975].

TRAVELNET (the particular semantic network used by the travel budget manager's assistant) is realized physically in three files [Woods et al., 1975]. The only one loaded into core is a 2500 word index array in which

each array location corresponds to a node. (The final version of TRAVELNET has 2476 nodes.) At each location there are either (1) file pointers into a second, randomly accessed file that contains the link-value pairs for the node, (2) pointers to list structure (in core) for the link-value pairs, or (3) node free-list indicators and pointers. Access to a node may be by its number (the array index) or (for nodes with names) by its name. In the latter case a third file containing name-number pairs is searched for the correspondence. A flag can be set to direct whether file information, once accessed, is to remain in core. For the index file, 16 pages of storage are used; for the list structure file, 82 pages; and for the name file, 8 pages. The maintenance of the network on external files eliminates a potential, serious space problem, without seriously impairing execution time. Furthermore, the system can be run so that nodes once accessed remain in core, requiring the cost of file access to be paid only once.

E.2 Command Language

The TBM assistant has a command language that can be used as a means of accessing the data base directly or as the language of semantic interpretations produced from the grammar. Expressions in the command language may be atomic, in which case there is an operator applied to arguments, or quantified compound expressions that use the FOR: quantification operator. The operators in atomic expressions specify operations to be performed on the data base or interactions with the user. The arguments may either refer to elements of the semantic network or be arbitrary constant expressions. In the first case, the argument may be specified by its print name, its index in the network, or by a LISP expression to be evaluated. Some examples of English sentences and their expression in the command language are shown in Appendix I.2. An explanation of some of the functions referenced there, as well as some others, is given below:

(ADD: node link value)

Adds value to node under the attribute link. If link has an ADDFN property associated with it, its value is a procedure to be executed to add value. Otherwise, SEMNET primitives are used to make the appropriate relational or property connections.

(BUILD: item-type (link1 value1) (link2 value 2) ...)

Builds an item which is an instance of item-type and has the specified link-value pairs. Uses ADD: for each pair and also adds DB/CREATOR and CREATE/TIME links.

(COMPLETE: item)

Special command which searches through item description and asks for missing values. It stops when the description is complete or when the user says "stop". COMPLETE: also allows the user to say "unknown" to any question.

(COMPUTE: node link)

A command to compute, as opposed to just find, a value for node and link; equivalent to (GET: node link ? DEFAULT). (See below.)

(FIND: (link1 value1) (link2 value2) ...)

Finds an item which has the specified link value pairs. Uses GET: for each pair. FIND: is an enumeration function which can be used with FOR: (see below) to produce elements one at a time.

(FOR: quantifier variable / class : restriction ; command)

Applies command to elements of class for which restriction holds and as determined by quantifier. Variable is bound to elements of the restricted class and is a free variable in command. (The permissible values for quantifier have been generalized from those in LUNAR [Woods, W. A., et al. 1972] to allow specification of cardinals by (THE <number>)).

(GET: node link value flag)

If value is NIL or ? then GET: follows link from node and returns value. Otherwise it verifies whether the specified value is stored. Flag determines the extent of search. If flag is T then no search is done and the explicit value stored, if any, is returned. If the flag is DEFAULT, then inferences are done as determined by METHODS associated with the link name (see E.5). If no METHOD succeeds, then the speaker is asked for help. If flag is FORCE-METHOD then the explicit value is ignored and METHODS must be used.

(OUTPUT: node syntactic-type length-of-response-flag)

Examines arg to determine appropriate form of printout, prints out an English-like description of arg, and if AUDIOFILE is set, produces a phoneme-prosodic cue list for speech generation.

E.3 Optimization

An interpretation produced by the parser using the pragmatic grammar is intended to be a procedural representation of the meaning of an utterance. The procedure is written in the system's own command language (see previous section), yet it cannot in general be executed directly. The reason is that the typical ways of referring in English do not map directly to efficient data base structures. Thus there is a need for a procedure to convert meaning representations into efficient, data base compatible procedures.

In the TBM assistant, the OPTIMIZE procedure is applied to each semantic interpretation before it is executed. It does three kinds of operations on the interpretation. First, it corrects for minor incompatibilities between the interpretation and data base structures and

procedures. These incompatibilities arise from changes in the grammar or semantic network. Reconciling them directly is not only troublesome, it also tends to lead to either inefficient structure building in the grammar or inefficient execution of interpretations.

The second kind of operation performed by OPTIMIZE is to recognize and accommodate reference to non-existent structures. Just as GET: (Section E.5) allows one to refer to fictitious links, OPTIMIZE allows one to refer to fictitious structures. For example, there are no instances of LOCATION stored in the same sense as instances of PERSON, CITY or DB/TRIP. Nevertheless one can write something of the form, (`-- (FINDQ: LOCATION --)` `--`) and have it executed properly. OPTIMIZE does this by (1) determining that there are no instances of LOCATION, (2) getting the FINDFN associated with LOCATION, and (3) replacing the (`(FINDQ: LOCATION --)` `--`) expression by a call to the FINDFN. In the case of LOCATION, the FINDFN does the appropriate search through instances of CITY, STATE, and COUNTRY.

The third operation performed by OPTIMIZE is to order the link-value pairs in each FIND: expression (see Section E.2). The order in which these pairs are checked can greatly affect execution time. For example, the expression,

```
(FIND: (INSTANCE/OF PERSON)
      (TRAVELER/OF A0046))
```

should not be executed by enumerating people and checking each to see if s/he is the traveler on trip A0046. Instead (since trips have only one traveler and there are many people), one should go to A0046 and find its traveler. Clearly, relations with single-valued inverses (such as TRAVELER/OF) should be checked before other relations. Similar considerations led to the ranking that OPTIMIZE uses:

- (1) relations with single-valued inverses
- (2) most other relations
- (3) properties (i.e. one way links that typically point to structures outside of the network)
- (4) "bad" link-value pairs, such as (TIME 'PAST), which do little filtering on the set of potential items

Our experience with OPTIMIZE suggests clearly that it is crucial to efficient execution, and that much remains to be done. For a discussion of

some possibilities, see [Reiter, 1975].

Examples of interpretations before and after optimization are shown in Appendix I.2. (The effect of the third OPTIMIZE operation is not shown because in ordering the link-value pairs numbers are substituted for node names so that the result is read easily only by computer).

E.4 Base Structures

E.4.1 Characterization

The data base organization for the TBM assistant accommodates a general budget management problem. In particular, the notion of "budget item" has been isolated as the basic combining element for a budget regardless of its purpose. A budget item represents an allocation of resources needed to carry out a plan and covers various activities pertaining to that plan (Figures 14 and 15).

For the travel data base the "allocation" of a budget item specifies those amounts of money initially allocated, currently allocated, spent and remaining. The "plan" for a budget item is a possibly vague description of one or more similar trips, such as "3 people to the ASA Meeting". The budget item then "covers" the actual trips which the manager associates with the plan.

The data base structures permit a distinction between a "proposed trip" and a "travel plan". In the first case the structure represents an actual trip, with a specific traveler, trip number, dates, etc., which has not yet been taken. In the second case the structure represents an expectation of the manager about a class of trips, usually without dates, and often specified only in terms of a number of travelers and a purpose and/or destination. Actual data from the BBN Speech Understanding project and a few related projects shows this to be a useful distinction. Some of the relationships among data base structures are shown in Figure 14-16.

An item representing an instance of a trip is shown in Figure 16. In the figure, the oval numbered 1172 represents the trip. The links (and other potential links) for this trip are described briefly below:

INSTANCE/OF: Points to DB/TRIP meaning that this item is a particular data base trip.

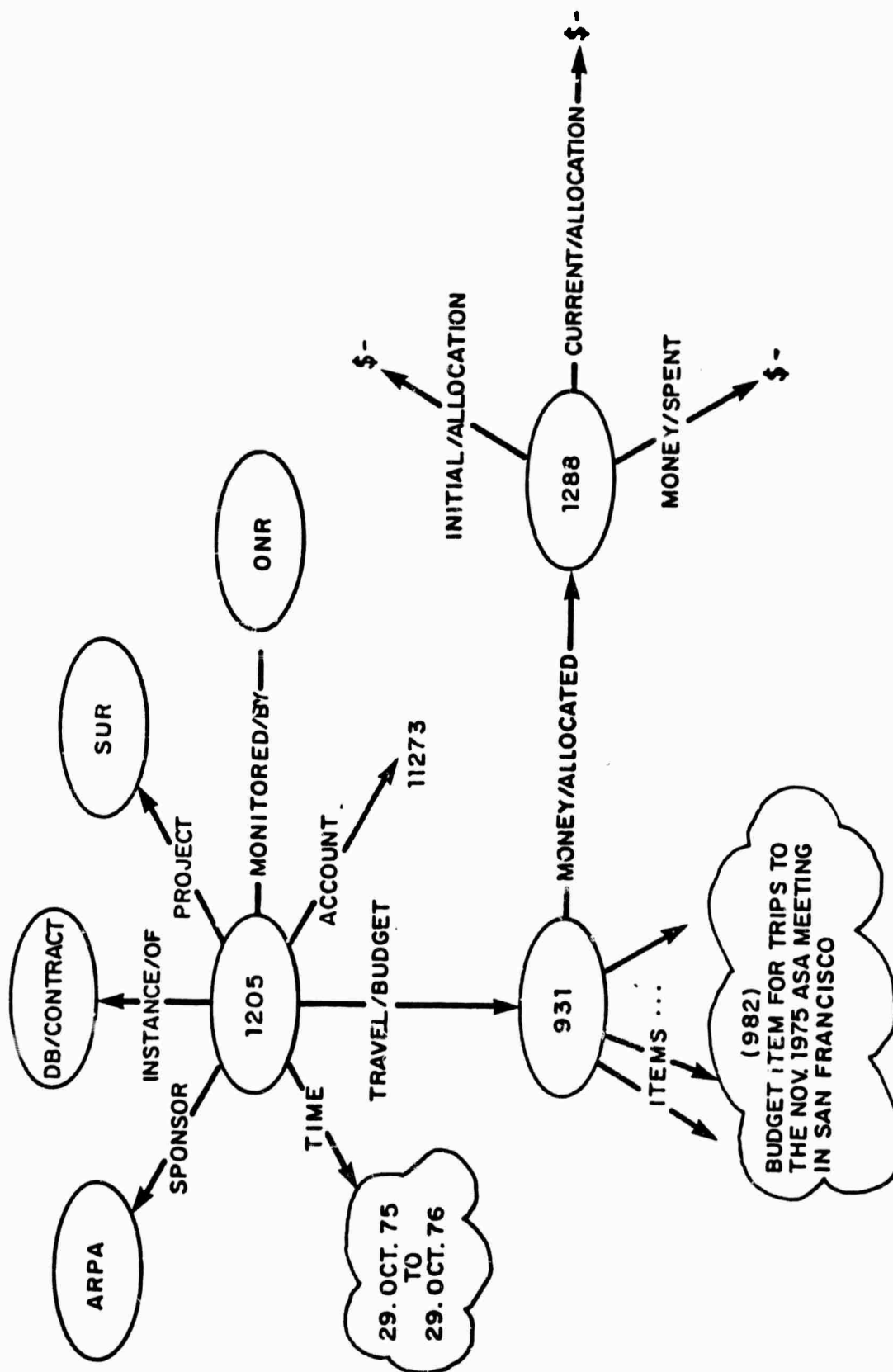


FIGURE 14. A CONTRACT AND A BUDGET

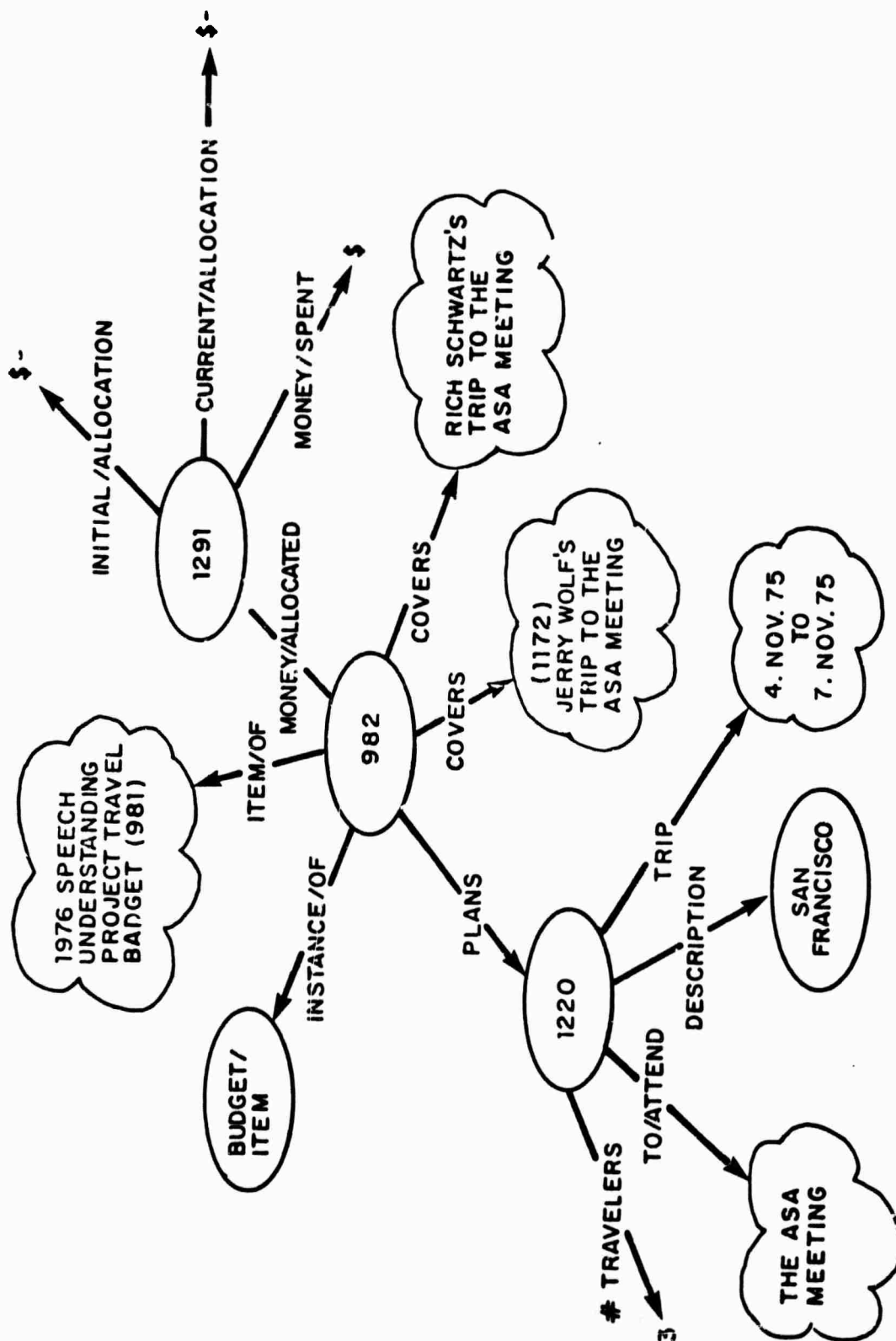


FIGURE 15. A BUDGET ITEM

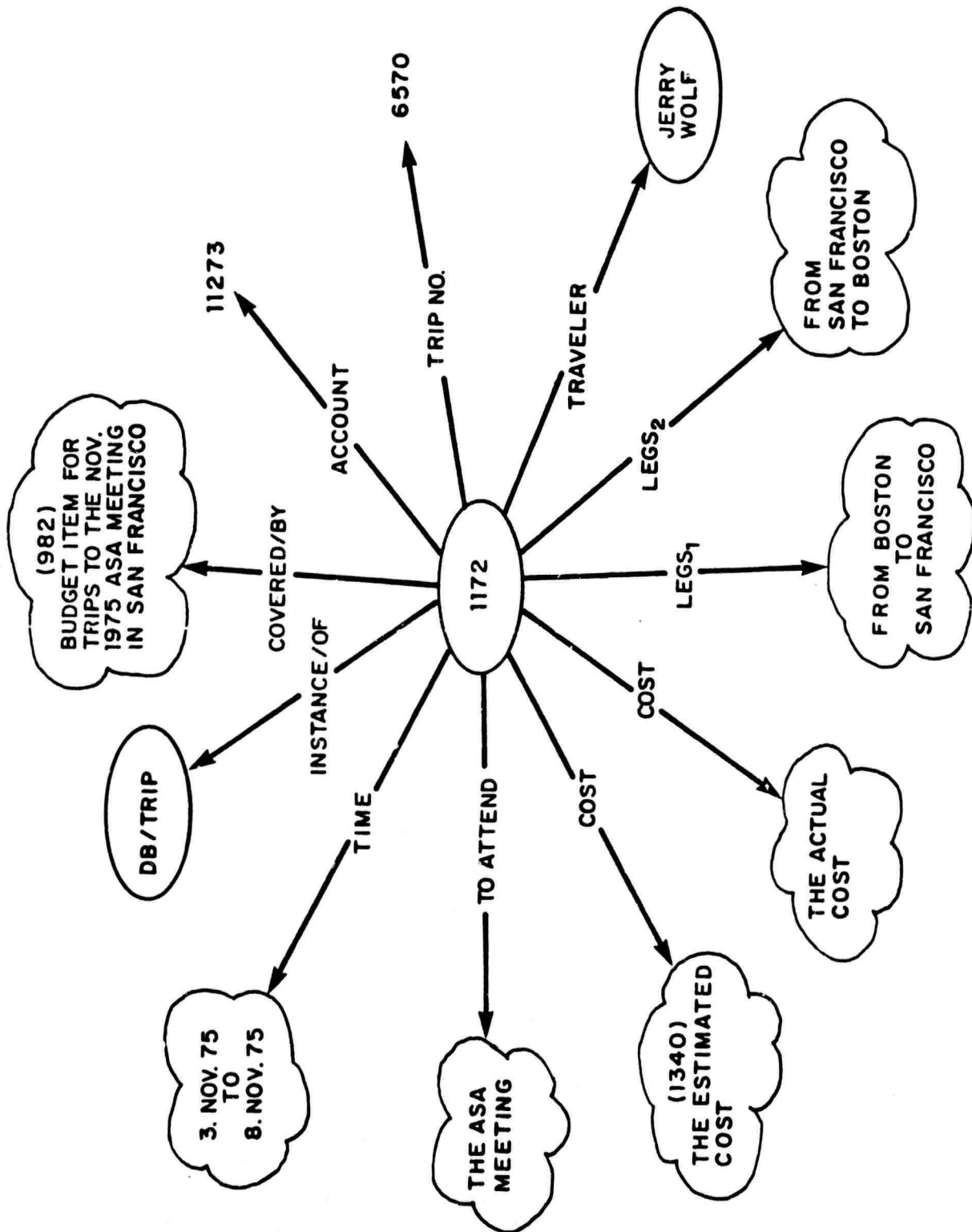


FIGURE 16. A TRIP

DB/CREATOR, CREATE/TIME: The speaker and time when this item is added to the data base. Items may also be added by the system as consequences of other events.

TRAVELER: Must be added by the speaker. A function, UNIQUE-REF, determines which person is meant if the traveler is incompletely specified.

#TRAVELERS: The number of travelers on the trip; especially important for travel plans where the traveler(s) may not be known.

STARTING/POINT, DESTINATION: Links to places.

TIME: Points to a time period, which is itself a structure with beginning and ending times and/or a duration.

MODALITY: May be "taken", "scheduled", "hypothetical", "ephemeral" (if created by the TBM assistant merely for the purpose of inference), or "unknown status."

COVERED BY: The BUDGET-ITEMs for which this trip is an item. Note that BUDGET/ITEMs are also items with structure.

LEGS: Trips are composed of "legs", each with a starting/point, a destination and a time.

TO/ATTEND: The conference (if any) which the traveler is attending.

TRIP#, ACCOUNT: Point to numerical values.

COST: Estimated and actual costs of the trip. Estimated costs may come from the manager or the system.

Considerations of parsing and interpretation have determined the existence of several data base structures. For example, the per diem associated with a city had been represented as a property (one-way link) of the city. In English, however, the per diem is usually referenced by a noun phrase as in "What is the per diem for Chicago?" The most natural interpretation results in a per diem structure, which itself has properties (e.g., a dollar value) and relations (e.g., an associated city).

E.4.2 Implementation

The data base is implemented in the SEMNET formalism (see Section E.1). Elements of the data base are items such as budgets, trips, fares, and dates (see Figure 17). Each item is an INSTANCE/OF some corresponding general concept (or type); e.g., a particular budget instantiates the general concept of a budget. Links specify the relationship of an item to other items and to specific information such as numerical values, names,

and places. The links from the type node define its use. For example, the link, LINKS, specifies those links that are normally associated with an instance. This is used in printing, searching, building and completing instance descriptions.

TRIPS

TRIPS	85
LEGS	59
COSTS	47
TIME PERIODS	85
TIME POINTS	65
	<hr/>
TOTAL	282

CONFERENCES

CONFERENCES	12
FEES	3
TIME PERIODS	12
TIME POINTS	10
	<hr/>
TOTAL	35

PEOPLE

PEOPLE	113
FIRSTNAMES	110
LASTNAMES	110
	<hr/>
TOTAL	333

BUDGETS

BUDGETS	7
BUDGET ITEMS	25
TRAVEL PLANS	19
CONTRACTS	6
CONTRACT MONITORS	2
PROJECTS	3
PROJECT SPONSORS	2
ALLOCATIONS	30
TIME PERIODS	12
TIME POINTS	15
	<hr/>
TOTAL	121

PLACES

CITIES	560
STATES	50
COUNTRIES	48
CITY PAIRS	30
FARES	40
	<hr/>
TOTAL	728

FIGURE 17. FACTUAL KNOWLEDGE IN TRAVELNET
(NUMBERS ARE APPROXIMATE NUMBER OF NODES FOR EACH TYPE)

Associated with each link name is an item that contains information true of all tokens of that link name. This allows us to organize general knowledge about how a link is to be used. For example, the storage of an

ISA link between two concepts implies an inverse, KINDS, link in the other direction. Therefore, the type nodes for ISA and KINDS have pointers to the inverse type. The set of possible values for a link are indicated by VALUE/CLASS. For example, the item for the link name, TRAVELER, has the VALUE/CLASS, PERSON.

A linkname item may also contain information specifying how to add (or remove) a link of that type. This is indicated by an ADDFN (adding function), or a REMOVEFN (removing function). For example, a DURATION link is not attached directly to the node for a trip but indirectly via a time period structure. The time period structure contains information such as the beginning and ending times of the time period, as well as the duration. Any or all of these pieces of information may be stored depending upon what the speaker has told the system. An ADDFN makes it possible to conceptualize duration as being associated with the trip or the time period, or with one of the legs of the trip. Since ADDFNs and REMOVEFNs may themselves cause additions or removals of links, there is often a cascade of structure changes upon assimilation of a new piece of information.

Finally, there is a METHOD associated with each linkname item. This is a procedure that is invoked whenever the value for a particular link of that type is needed but missing. The ultimate default is to ask the speaker. These METHODS are discussed in the next section.

E.5 Inference

Inference in the system (see also [Bruee and Harris, 1975]) can be viewed as a natural generalization of the notion of structures with slots and default values for each slot. Here, instead of being values, defaults are procedures (called METHODS) for determining the appropriate value whenever a slot filler is missing. The procedures may in turn request other slot values, which may require activating other default procedures.

The inference process is implemented via the function, GET:. GET: can be used to find the value for a node-link pair or to verify that a specified value is there. A flag can be set that determines the depth of search and whether or not the speaker is to be asked in the event of failure. The effect of the GET: implementation is that the basic operation

of requesting the value of an attribute of an object is not conditioned by (perhaps arbitrary) data base constructions. It also means that whereas a new attribute must be structurally defined, there does not need to be a special set of functions for retrieving its value under an indefinitely large assortment of situations. In effect, one can define arbitrary "fictitious links" (Figure 18) and use them as actual links.

For example, the call (GET: <trip> 'cost) could be part of the interpretation of "Estimate a cost for that trip." If cost were stored explicitly, then no deduction would be required. If not, a cascade of calls to GET: could result using what explicit information there happened to be (Figure 19). The recursive mechanism of GET: is driven by the property METHODS on the link name specified in the call to GET:. Each METHOD consists of an APPLICABILITY/TEST which restricts the application of the method, a FUNCTION naming an operation to be performed, a METHOD/RANK which gives the priority of the method relative to other methods for the same linkname, and ARGUMENT/PATHS which specify, for each argument of the FUNCTION, what links to follow (via GET:) from the present node to get the desired values. Methods are applied in order (by METHOD/RANK) until one succeeds or all fail. As each method is applied, it builds a trace of its computation tree such as that shown in Figure 20. In principle, the tree enables the system to answer the question "How did you get that?" or to set monitors on questions about trivially different computations, e.g., "What if the per diem were twenty dollars?", or "What if the trip lasted six days instead of four?", etc.

E.6 Discourse Model

Early simulations of dialogues between a travel budget manager and the computer assistant suggested a discourse model in which, at any point, the user could be seen as being in one of several states. S/he could be trying to determine the consequences of a proposed trip, or examining the state of the budget, or entering new trip information. We have considered several formulations of a discourse model in which these states would be made explicit and used to advantage.

One such formulation has involved the concepts of modes of interaction and intents [Woods, et al., 1974, pp. 201-232; Bruce, 1975a]. An intent

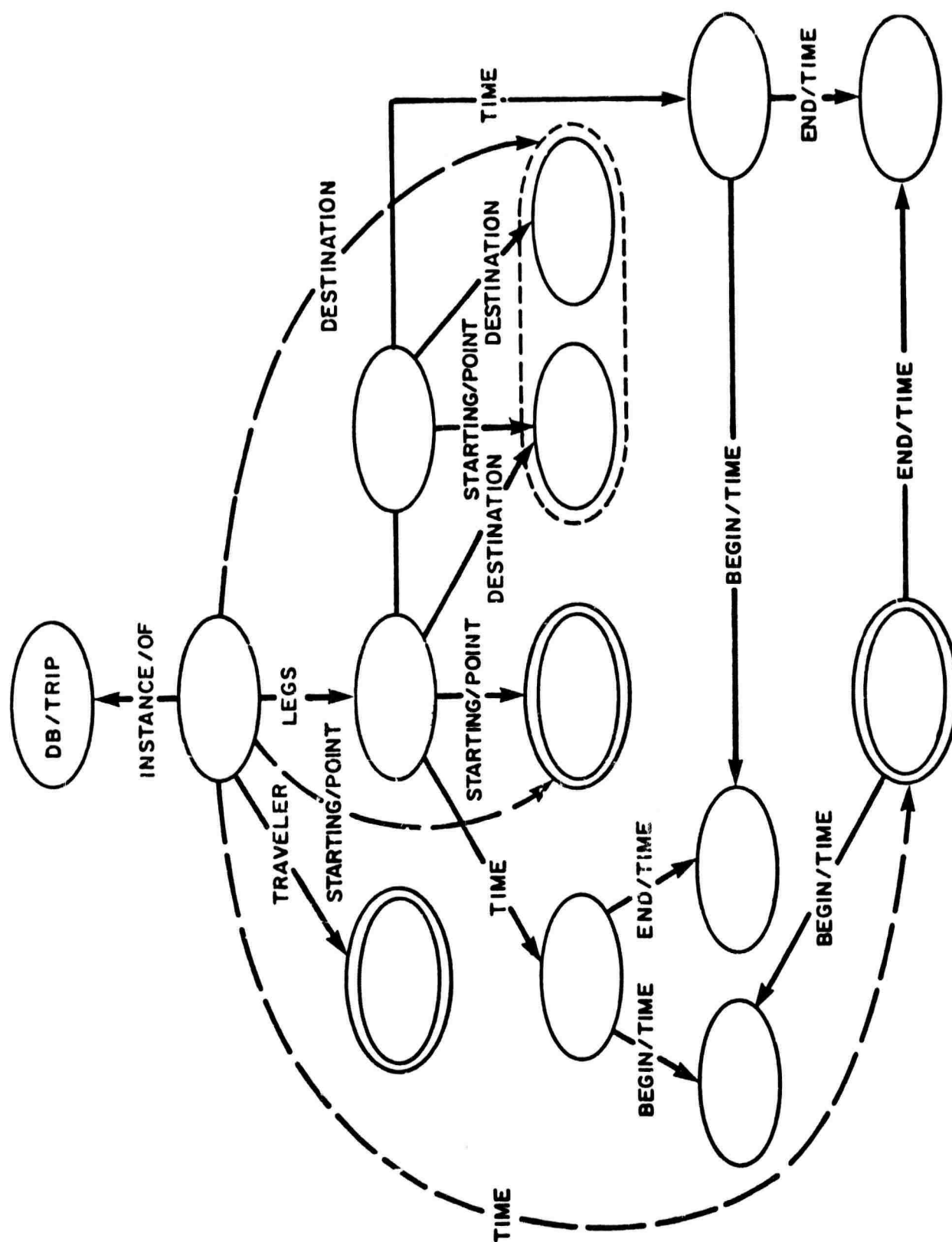


FIGURE 18. FICTITIOUS LINKS

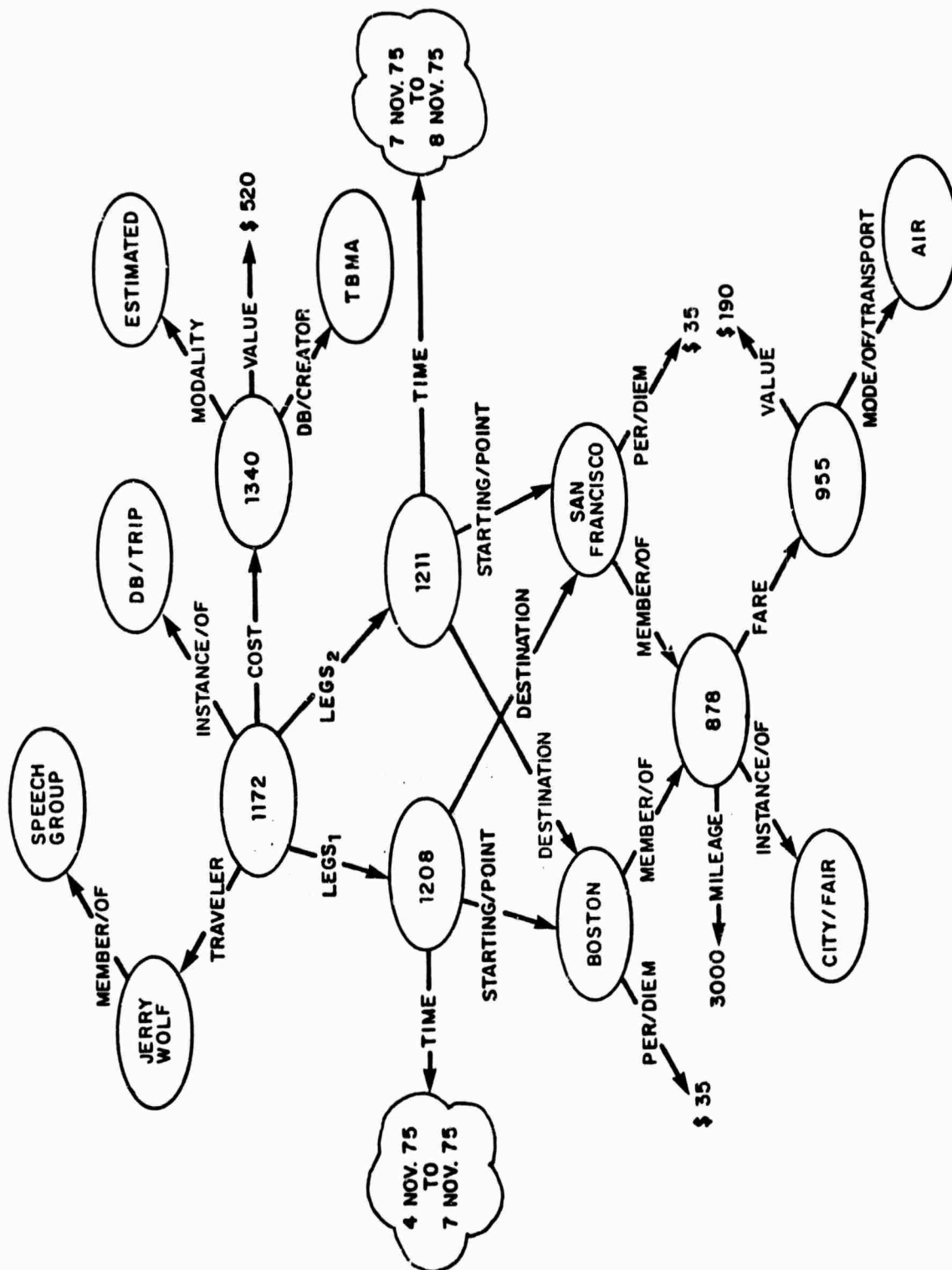


FIGURE 19. INFORMATION USED IN ESTIMATING THE COST OF A TRIP

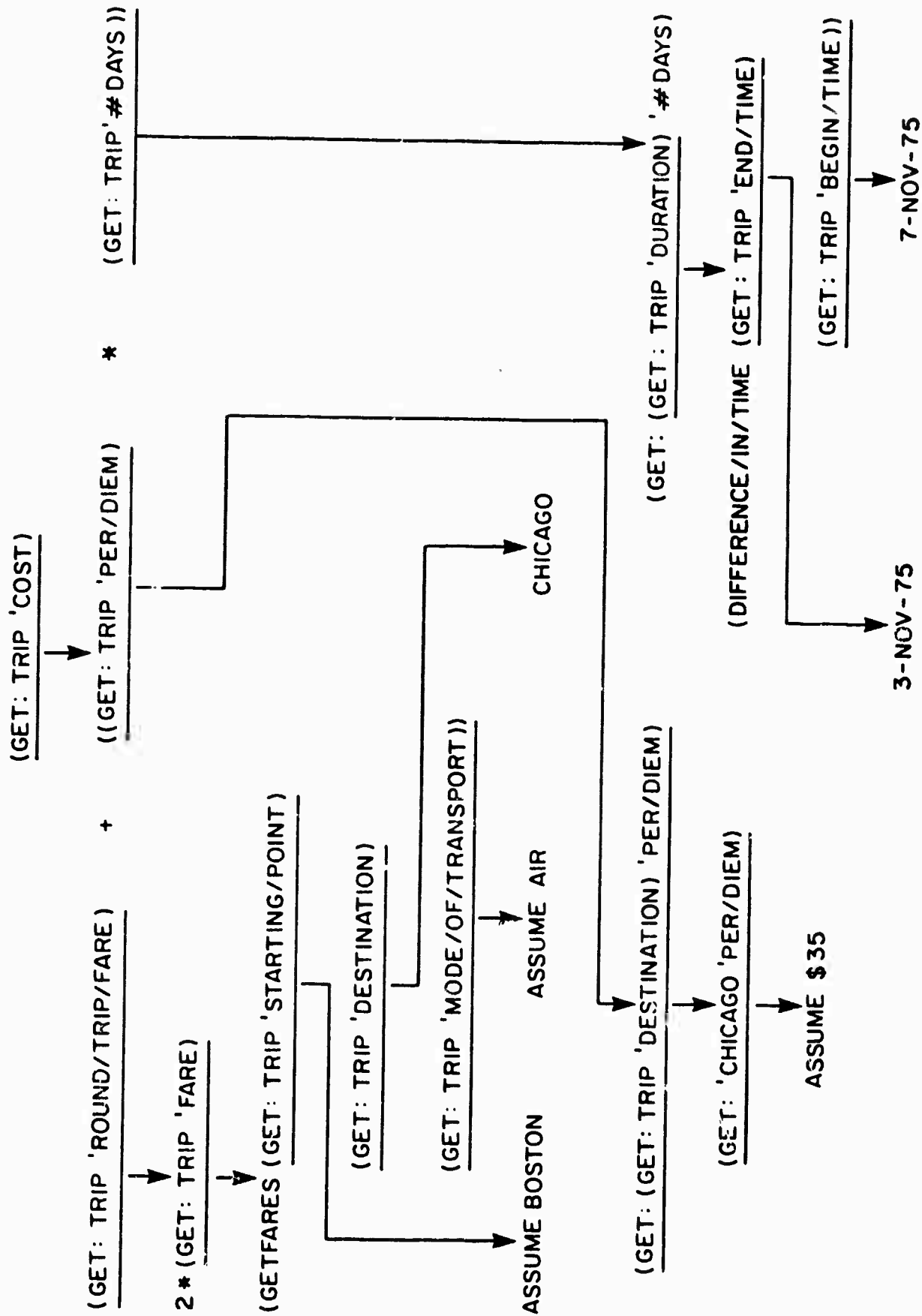


FIGURE 20. TRACE OF A COST COMPUTATION SHOWING USE OF METHODS

is the assumed purpose behind an utterance. Patterns of intents such as,

- user-enter-new-information
- system-point-out-contradiction
- user-ask-question
- system-answer-question
- user-make-editing-change
- system-confirm-change,

constitute the modes of interaction.

An augmented transition network (ATN) grammar was used to represent some of the common modes of interaction found in travel budget management dialogues. Then a modified ATN parser was written that steps through the grammar on the basis of the input sentence structure and the then-current state of the data base. At any given state the parser can predict the most likely next intent and hence such things as the mood and head of the next utterance. This model was not used in the final version of the system because of several limitations. Primarily it was too rigid to function alone as a discourse model (but see [Bruce, 1975]). Also we found it difficult to characterize and apply for speech understanding the linguistic consequences of a discourse state.

Another formulation of the user/discourse model involves the notion of demands and counter-demands made by participants in the dialogue. These include such things as unanswered questions and contradictions which have been pointed out. (Both of these formulations are discussed more fully in [Bruce, 1975b].

This latter formulation allows us to model how one computation of a response can be pushed down, while a whole dialogue takes place to obtain missing information, and how a computation can spawn subsequent expectations or digressions. Some elements of this demand model, together with other aspects of the implemented discourse model, are explained below:

(a) Demands: These are demands for service of some sort made upon the system by the user or by the system itself. An active unanswered question is a typical demand with high priority. The fact that some questions cannot be answered without more information leads to the kind of embedding called a "mode of interaction" above. Demands of lower priority include such things as a notice by the system that the manager is over his budget. Such a notice might not be communicated until after direct questions had been answered.

(b) Counter-demands: These are questions the system has explicitly or implicitly asked the user. While it should not hold on to these as long as it does to demands, nor expect too strongly that they will be met, the system can reasonably expect that most counter-demands will be resolved in some way. This is an additional influence on the discourse structure.

(c) Current topic: This is the active focus of attention in the dialogue (see Figure 21). It could be the actual budget, a hypothetical budget, a particular trip, or a conference. The current topic is used as an anchor point for resolving definite references and deciding how much detail to give in responses. Again, this structure leads to certain modes of interaction. For example, if the manager says "Enter a trip," the system notes that the current topic has changed to an incompletely described trip. This results in demands that cause standard fill-in questions to be asked. If the manager wants to complete the trip description later, then the completion of the trip description becomes a low priority demand.

(d) Miscellaneous deictic structures: The discourse area of the data base also contains an assortment of items strongly linked to the here and now, including:

- a) NOW, a pointer to the current time and date,
- b) SPEAKER, a pointer to the current speaker,
- c) the last mentioned person, place, time, trip, budget, conference, etc.

The system has a primitive one-queue implementation of the "demand model." This queue consists of forms such as (DO (FOR: --) --) which represent the speaker's previous queries and commands as well as commands initiated by the system to examine the consequences of its actions, give information to the user, or check for data base consistency. These forms are related by functional dependencies and relative priorities. The implemented demand types are:

DO: means execute the specified command.

TEST: means evaluate the form and answer "no" if NIL or "yes" otherwise.

RESPOND: means give the user some information (which may or may not be part of an answer to a direct query).

PREVENT: means monitor for a subsequent possible action and block its normal execution (as in "Do not allow more than three trips to Europe.").

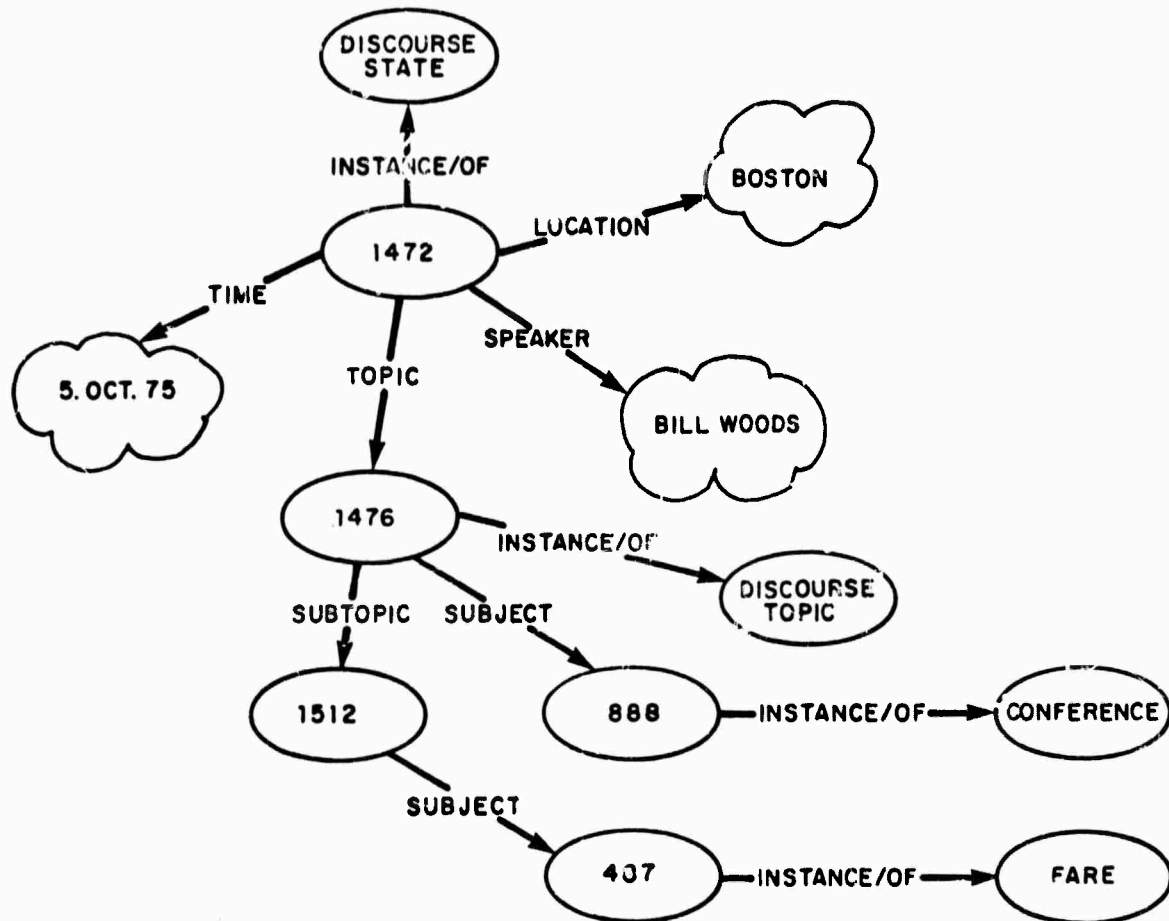


FIGURE 21. A DISCOURSE STATE

F. Response Generation

F.1 Overview

Once a satisfactory theory for an utterance has been constructed, it must be acted upon by the system. Regardless of the type of action taken, some appropriate response should also be made. From the speaker's point of view the response should be explicit, concise, and easy to understand. Its role in the speaker-system interaction is complex in that it may affect the speaker's way of describing entities in the domain and can tell him/her much about the capabilities of the system and how it operates.

A text response generation program (Section F.2) and a speech synthesis-by-rule program (Section F.3 and Volume 2, Section C) allow the TBM assistant to speak to the manager in an English-like language. For example, it may describe a trip as:

John Makhoul went to Pittsburgh from Monday, the 30th of June, to Wednesday, the 2nd of July, 1975.

This text string is then converted into a string of phonemes and special symbols (stress marks, word and phrase boundary markers, etc.), and passed to the synthesis program which produces a waveform file. Such a response generation capability also serves as a means of verifying phonemic spellings in the dictionary and testing the synthesizer's performance. The next section presents some examples based on actual data in TRAVELNET and discusses the functions currently used in producing English responses describing them.

F.2 Text Response Generation

The top level program for response generation is OUTPUT:, a function that produces both the text response and the phoneme-prosodic use string for speech synthesis. OUTPUT: takes a semantic network node, a constituent type and a flag indicating the desired length of the response. The node should be an INSTANCE/OF a general type (e.g., a particular trip is an INSTANCE/OF the concept of DB/TRIP). Associated with the type node is a set of language generation frames representing the base forms of syntactic constituents for describing the instance nodes. Depending upon the response length flag, the appropriate syntactic constituent, and other possible conditions. OUTPUT: selects a language generation frame and fills

it in with information from the semantic network. The filled frame is linearized and printed out. Transformations may be applied, for example, to change the verb form from present to past. If the AUDIOFILE flag is set OUTPUT: also produces a phoneme string with prosodic markers (see Section F.3). This section shows examples of nodes in the semantic network, generation frames, generated English text, and generated phoneme-prosodic cue strings.

The first item shown below is the type node for conferences. Note that there are 12 items (462 579 888 --) that are in the INSTANCE/OF relation to DB/CONFERENCE. Also note that there are 3 response generation frames signified by the G-FRAMES relation. What follows shows how a particular G-FRAME for DB/CONFERENCE gets instantiated with respect to a particular instance of DB/CONFERENCE, and is subsequently used in generating a response.

637 - DB/CONFERENCE

CREATOR	CHIP
TYPE	LINKNAME
ALIAS	DB/MEETING
LINKS	(TIME) (LOCATION) (SPONSOR) (CREATE/TIME)
	(DB/CREATOR) (ATTENDED/BY) (REGISTRATION/FEE)
INSTANCES	462 579 888 961 965 972 976 1029
	1034 1041 1251 1256
METHODS	[Follow back pointers from time point to
	db/conference 956]
VALUE/CLASS/OF	(TO/ATTEND) (REGISTRATION/FEE/OF)
ENGLISH/NAME	(CONFERENCE)
SYNTAX/CLASS	(MEETING)
INSTANCE/OF	(LINKNAME)
G-FRAMES	1413 1414 1415
ACTION	[WILL BE HELD 2471]
LAST/MENTIONED	462

A typical instance of DB/CONFERENCE describes a recent ACL Meeting. It has a LOCATION, TIME, and other information that can be used in describing it, depending upon such things as the desired length of response and the desired focus.

462

CREATOR	CHIP
INSTANCE/OF	(DB/CONFERENCE)
SPONSOR	(ACL)
LOCATION	[BOSTON MASSACHUSETTS 154]
TIME	546
DB/CREATOR	[CHIP BRUCE 276]
REGISTRATION/FEE	901
LAST/MENTIONED/OF	(DB/CONFERENCE)

There are, as mentioned above, 3 response generation frames or G-FRAMES for DB/CONFERENCE. They correspond to the syntactic constituents S, NP and PP. In order to describe node 462 in English we have to fill one of these G-FRAMES. Suppose we want to describe it with a complete sentence.

The call to OUTPUT:, giving the node to be described, the desired constituent type and the response length flag, might be

```
(OUTPUT: 462 'S 'MEDIUM)
```

Let us consider first how to fill the G-FRAME used to describe node 462 as an S (see Appendix I.4 for the other G-FRAMES):

1413

```

FRAME          (NP SELF)
                (V ACTION)
                (PP LOCATION *OPT*)
                (PP TIME (OMIT TIME/OF) *OPT*)
CREATOR        CHIP
TYPE           S

INSTANCE/OF    (G-FRAME)
G-FRAME/OF     (DB/CONFERENCE)

```

The central thing to notice here is the FRAME link. It points to a structure made up of one or more slots. Each slot is labeled with a constituent type, a filler, and zero or more flags and qualifiers. For example, the first slot in G-FRAME 1413 is (NP SELF). The constituent type is NP and the filler is SELF. This means that in order to fill G-FRAME 1413, one must first follow the SELF link from the item being described and fill a G-FRAME of TYPE NP using the item at the end of the SELF link. SELF happens to be a special link which is always computed by a METHOD that returns the item itself, i.e. (GET: <item> 'SELF) => <item>.

A more interesting case is the slot, (PP TIME (OMIT TIME/OF) *OPT*). The constituent type is PP and the filler is TIME, meaning, find the TIME of the node being described and describe it as a PP. However, there are also two qualifiers on this slot. The first, (OMIT TIME/OF), says that when the node at the end of the TIME link is being described, its TIME/OF should not be described. This is to prevent sentences like, "The ACM conference was held ... from the beginning time of the ACM conference to the ending time of the ACL conference." The second qualifier, *OPT*, says that this slot is optional, i.e. if there is no TIME for the node, do not abort the filling of this G-FRAME. (In general, aborting does not mean

that no description can be generated, but simply that another G-FRAME must be tried. If all G-FRAMES fail, the node is printed in tabular form.) In this case there is a TIME for node 462. It is represented in the following three nodes:

546

CREATOR	CHIP
INSTANCE/OF	(TIME/PERIOD)
BEGIN/TIME	461
END/TIME	536
TIME/OF	462
DURATION	991

461

HOURL	9
CREATOR	CHIP
YEAR	1975
MONTH	10
DAY/OF/MONTH	30
INSTANCE/OF	(TIME/POINT)
BEGIN/TIME/OF	546
PRECEDES	536
PRECEDED/BY	1105

536

HOURL	12
CREATOR	CHIP
YEAR	1975
MONTH	11
DAY/OF/MONTH	1
INSTANCE/OF	(TIME/POINT)
END/TIME/OF	546
PRECEDED/BY	461
PRECEDES	1209

The node at the end of the TIME link in our example DB/CONFERENCE is a TIME/PERIOD node (546). In order to describe it as a PP we need to consider the four G-FRAMES for TIME/PERIOD whose types are PP. (All the G-FRAMES for TIME/PERIOD are shown in Appendix I.4):

1448

PRECONDITION	(NOT (POINT-TIME? ITEM))
FRAME	(PP BEGIN/TIME (USE "from"))
	(PPY END/TIME (USE "to"))
CREATOR	CHIP
TYPE	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/PERIOD)

1449

FRAME	(PP BEGIN/TIME)
CREATOR	CHIP
TYPE	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/PERIOD)

1450

FRAME	(PP END/TIME)
CREATOR	CHIP
TYPE	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/PERIOD)

1451

FRAME	(PP DURATION (USE "for"))
CREATOR	CHIP
TYPE	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/PERIOD)

Taking these G-FRAMES in the order presented, OUTPUT: first checks the PRECONDITION, if any, and then tries to fill the G-FRAME. Node 546 represents a TIME/PERIOD which is not a single day; for this domain that means the PRECONDITION, (NOT (POINT-TIME? ITEM)), in the first G-FRAME, succeeds. The resulting candidate G-FRAME (node 1448) has two slots to be filled, one for the BEGIN/TIME and one for the END/TIME of the TIME/PERIOD. Once again the process of filling G-FRAMES is invoked, this time for TIME/POINTS as PPs.

The recursive process of filling G-FRAME slots continues until a slot is reached that has either a string filler, e.g. (PREP "to"), or the slot is a property type link that points to structures outside the network, e.g. (ADJ VALUE), where VALUE points to a number.

Other qualifiers on a G-FRAME slot are of the form (USE <string>) and (OMIT <link>). Both are used to communicate from one G-FRAME to those it invokes. The USE qualifier takes effect when there is a slot filler in the lower G-FRAME of the form (DEFAULT <string>). In that case the string in the USE expression is inserted in place of the default string. (OMIT <link>) is used by the higher G-FRAME to prevent the filling of any slot in the lower G-FRAME that has <link>. This works if the lower slot is marked *OPT*; otherwise, it aborts the lower G-FRAME.

We can summarize the G-FRAME structure as follows: A G-FRAME is defined by a (possibly null) PRECONDITION, a (syntactic) TYPE, and a FRAME consisting of a number of slots. Each slot has itself a type (that should match some G-FRAME TYPE), a filler and zero or more qualifiers. A filler can be a string, a linkname, a (DEFAULT <string>) expression, or a (FUNCTION <function> <link>) expression. In the latter case, <function> is

applied to the value at the end of <link>. (This is useful for such things as date expressions where a perpetual calendar routine must be invoked). Finally, qualifiers can be any of *OPT*, indicating the slot is optional; (USE <string>), indicating that <string> should be used at the lower level; or (OMIT <link>), indicating that the lower level slot(s) using <link> should not be filled. The final effect of our call,

(OUTPUT: 462 'S 'MEDIUM)

is to print out the text string

The October 1975 ACL CONFERENCE WAS HELD in BOSTON MASSACHUSETTS
from October 30th, 1975 to November 1st.

corresponding to the list of dictionary entries

(THE OCTOBER NINETEEN-M SEVENTY FIVE ACL CONFERENCE WAS HELD IN
BOSTON, MASSACHUSETTS FROM OCTOBER THIRTIETH NINETEEN-M SEVENTY
FIVE TO NOVEMBER FIRST)

and to produce the phoneme-prosodic cue list:

(FW # DH AH !2 # AX !- K * T OW !2 * B AXR !- # N AY !2 N * T IY
!1 N # S EH !2 * V AX !- N * T IY !0 # F AY !2 V # EY !1 * S IY
!1 * EH !2 L # K AA !2 N F * R AX !- N S # W AH !2 Z # HH EH !2 L
D # FW # IH !2 N # B AO !2 * C T AX !- N # M AE !1 * S AX !- * CH
UY !2 * S IX !- T S # FW # F R AH !2 M # AX !- K * T OW !2 * B
AXR !- # TH ER !2 * T Y IX !- TH # , # W AH !2 N # N AY !2 N # S
EH !2 * V AX !- N F AY !2 V - FW # T UW !2 # N OW !0 * V EH
!2 M * B AXR !- F ER !2 S T %.)

The next example shows the semantic network node for "fees" (DB/FEE) and an instance of DB/FEE (see Figure 22). Following that are two G-FRAMES for DB/FEE and an example response.

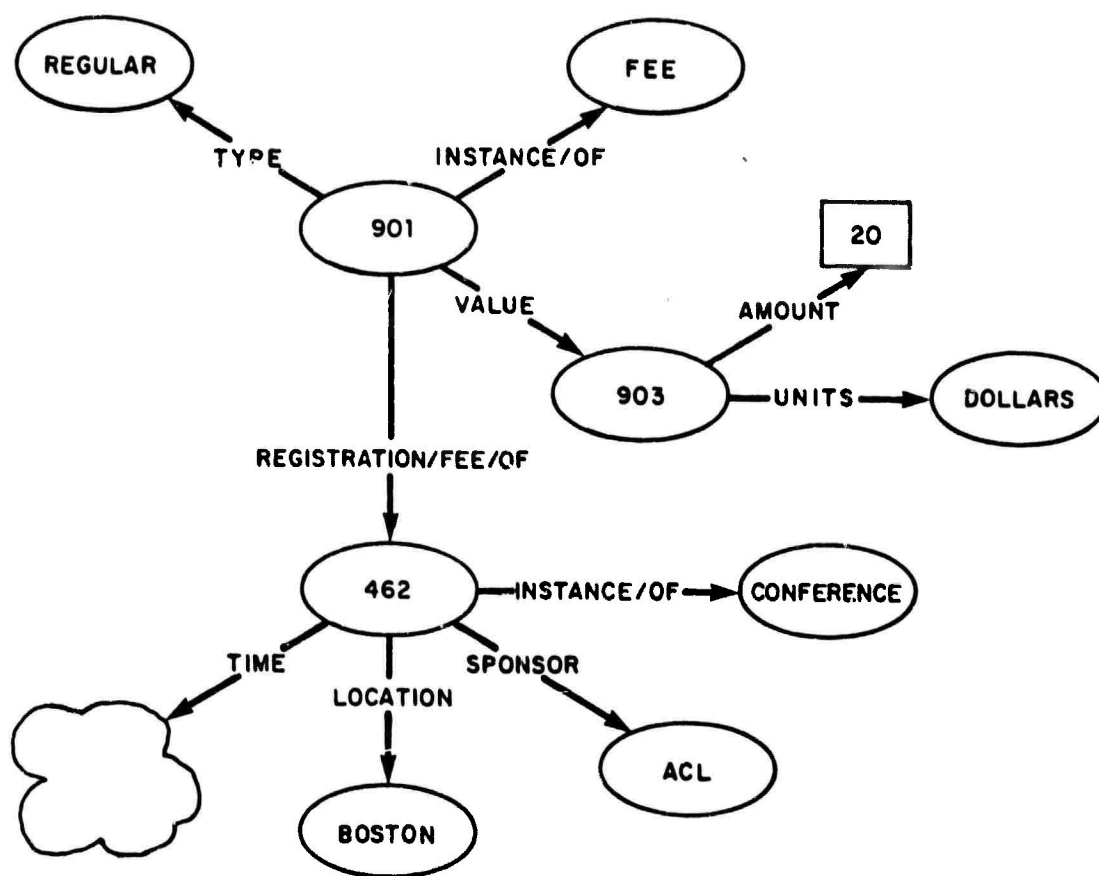


FIGURE 22. A REGISTRATION FEE

901

CREATOR	CHIP
VALUE	20
INSTANCE/OF	(DB/FEE)
DB/CREATOR	[CHIP BRUCE 276]
REGISTRATION/FEE/OF	462
CREATE/TIME	957
UNITS	(DOLLARS)
LAST/MENTIONED/OF	(DB/FEE)

1423

FRAME	(DET "The") (ADJ "registration") (N TYPENAME) (PP REGISTRATION/FEE/OF (USE "for"))
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/FEE)

1424

```

FRAME          (NP SELF)
                (V ACTION)
                (ADJ VALUE)
                (N UNITS)
CREATOR        CHIP
TYPE           S

INSTANCE/OF    (G-FRAME)
G-FRAME/OF     (DB/FEE)

```

(OUTPUT: 901 'S) produces the following English text string

The registration fee for the ACL conference is 20 DOLLARS.

corresponding to the list of dictionary entries

(THE REGISTRATION FEE FOR THE ACL CONFERENCE IS TWENTY DOLLAR-S)

and produces the phoneme-prosodic cue list

```

(FW # DH AH !2 # R EH !1 * JH IX !- * S T R EY !2 * SH EN !- # F
IY !2 # FW # F AO !2 R # DH AH !2 # EY !1 * S IY !1 * EH !2 L # K
AA !2 N * F AXR !- * EN !- S # IH !2 Z # T W EH !2 N * T IY !0 #
D AA !2 * L AXR !- Z %.)

```

The next example shows semantic network nodes for "fare" (DB/FARE) with an instance of a DB/FARE. This is followed by the G-FRAMES for DB/FARE and an example response. Note the use of the *OPT* flag in the frame to indicate that the mode of transportation is optional, and the (USE <string>) expressions to determine specific lexical entries.

485 - DB/FARE

```

BUILDFN        COMPLETE:

LINKS           (TYPE) (MODE/OF/TRANSPORT) (CREATE/TIME)
                (DB/CREATOR) (VALUE) (FARE/OF)
INSTANCES      159 477 798 949 950 951 952 953
                954 955 1054 1069 1070 1072 1073
                1075 1077 1078 1079 1081 1083 1085
                1087 1091 1093 1095 1096 1097 1098
                1099 1100 1135 1155 1157 1159 1174
                1335 1360
ENGLISH/NAME    (FARE)
PROMPTLINKS     (TYPE) (MODE/OF/TRANSPORT) (VALUE)
                (FARE/OF)
ISA             (ACCOUNTING/CONCEPT) (DB/EXPENSE)
G-FRAMES        1421 1422
LAST/MENTIONED  1095

```

1095

CREATOR LAURA
 VALUE 41

INSTANCE/OF (DB/FARE)
 DB/CREATOR [LAURA GOULD 501]
 CREATE/TIME 957
 MODE/OF/TRANSPORT (TRAIN)
 FARE/OF 555
 UNITS (DOLLARS)
 LAST/MENTIONED/OF (DB/FARE)

1421

FRAME (DET "The")
 (ADJ TYPE)
 (ADJ MODE/OF/TRANSPORT *OPT*)
 (N TYPENAME)
 (PP FARE/START (USE "from"))
 (PP FARE/END (USE "to"))

CREATOR CHIP
 TYPE NP

INSTANCE/OF (G-FRAME)
 G-FRAME/OF (DB/FARE)

1422

FRAME (NP SELF)
 (V ACTION)
 (ADJ VALUE)
 (N UNITS)

CREATOR CHIP
 TYPE S

INSTANCE/OF (G-FRAME)
 G-FRAME/OF (DB/FARE)

(OUTPUT: 1095 'S) produces the English text string

The ONE-WAY TRAIN FARE from BOSTON MASSACHUSETTS to PITTSBURGH
 PENNSYLVANIA IS 41 DOLLARS.

Corresponding to the list of dictionary entries

(THE ONE-WAY TRAIN FARE FROM BOSTON, MASSACHUSETTS TO PITTSBURGH,
 PENNSYLVANIA IS FORTY ONE DOLLAR-S)

and produces the phoneme-prosodic cue list.

(FW # DH AH !2 # W AH !2 N # W EY !2 # T R EY !2 N # F EH !2 R #
 F R AH !2 M # B AO !2 # S T EN !- # M AE !1 # S AX !- # CH UY !2
 # S IX !- T S # T UW !2 # P IH !2 T S # B ER !1 G # P EH !1 N # S
 AX !1 L # V EY !2 # N IY !0 # AX !- # IH !2 Z # F AO !2 R # DX IY
 !0 # W AH !2 N # D AA !2 # L AXR !- Z %.)

F.3 Speech Synthesis

Speech synthesis can be invoked in two principal ways. In one, the
 function CTALKER takes a typed-in English sentence which it converts into

HWIM dictionary format using the function SPEECHIFY (see Section D.2.1). It then produces a phonetic transcription of the sentence using the most likely pronunciation for each of the words (the pronunciation of highest probability in the expanded phoneme dictionary) and calls the speech synthesis program to generate a waveform file corresponding to the phonemes.

In the other mode, the function OUTPUT: takes a concept in the semantic network, a constituent type and a flag indicating desired length of output, and instantiates one of the frames to the concept (as described in the preceding section). OUTPUT: then linearizes the resulting parse tree, applying a few special purpose transformations e.g., tense changes on the verb. (A side effect of linearization is the printing of a text response.) As it linearizes the parse tree, OUTPUT: inserts prosodic cues for the speech synthesis program. For example, it inserts the symbol "FW" to indicate reduced stress in a following function word, and the symbol ")N", to indicate a major phrase boundary. These cues derive from the syntactic structure of the output, which is the reason for not working solely with the text string. From there the action is similar to that of CTALKER.

In the synthesis program, a mapping program translates the phonetic spellings together with the syntactic markers (non-phonetic symbols) from the phoneme set of the phonetic dictionary into that of the synthesis program. The syntactic markers, such as the "FW" and ")N", are used to construct the appropriate prosodic characteristics. The synthesis program uses the input string to produce a waveform file that can be played out as an audio response. Details of this program are discussed in Volume 2.

Work on response generation has benefited other components of HWIM in several ways:

- 1) We uncovered several errors in the phonetic dictionary having to do with spellings and pronunciation likelihoods. Missing words were also discovered (i.e., "way" as in "one way fare").

- 2) Audio response provided an impetus to exercise all aspects of the retrieval and inference routines which drive the audio generation. This provided an important additional check on data base consistency and

correctness.

3) We also did extensive testing of the synthesis-by-rule program (critical to the operation of the VERIFIER component). By using the travel budget dictionary, we tested those words and phonetic contexts which appear as spoken input to HWIM. Problems of pronunciation appeared and resulted in changes to the synthesis program. Certain acoustic-phonetic and phonological rules also required modification.

G. Conclusion

This report has presented some of the characteristics of the TBM assistant as it exists in the Fall of 1976. These characteristics reflect the goal of natural communication between a person and a computer assistant. Speech understanding (Volumes 1-4) has, of course, been the major part of the effort, but, as discussed here, work on response generation, discourse modeling, inference and data base design has also been important.

Much remains to be done on the system. Ultimately natural communication depends upon intelligence in both communicants, and the intelligence in the TBM assistant is of a very low order. Still, these programs and the limitations in ideas that they expose suggest many fruitful areas of further work.

H. References

- [1] Bates, M.A. and Bruce, B.C. (1975) "Time Expressions" in Woods, et al., 1975c.
- [2] Bobrow, D.G. and Collins, A. (1975) Representation and Understanding: Studies in Cognitive Science, Academic Press, New York.
- [3] Bobrow, R.J. and Brown, J.S. (1975) "Systematic Understanding: Synthesis, Analysis and Contingent Knowledge in Specialized Understanding Systems."
- [4] Brown, J.S., Burton, R.R. and Bell, A.G. (1974) "SOPHIE: Final Report," BBN Report No. 2790.
- [5] Bruce, B.C. (1975a) "Pragmatics in Speech Understanding." Proc. Fourth International Joint Conference on Artificial Intelligence, Tbilisi.
- [6] Bruce, B.C. (1975b) "Discourse Models and Language Comprehension." American Journal for Computational Linguistics, Microfilm 35, pp. 19-35.
- [7] Bruce, B.C. and Harris, G. (1975) "Procedural Semantics in the Travel System," in Woods, et al., 1975c.
- [8] Cook, C., Bruce, B.C. and Gould, L. (1975) "Audio Response Generation," in Woods, et al., 1975d.
- [9] Nash-Webber, B. and Bruce, B.C. (1976) "Evolving Uses of Knowledge in a Speech Understanding System." Proc. COLING 76, Ottawa. Also in Woods, et al., 1976b.
- [10] Nash-Webber, B.L. (1975) "SEMNET - The Network Utility Package" in Woods, et al., 1975a.
- [11] Reiter, R. (1976) "Query Optimization for Question-Answering Systems," Proc. COLING-76, Ottawa.
- [12] Shapiro, S.C. (1971) "A Network Structure for Semantic Information Storage Deduction and Retrieval," Proc. Second International Joint Conference on Artificial Intelligence, London.
- [13] Winograd, T. (1971) "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language," Project MAC Report, TR-84, MIT.
- [14] Woods, W.A., Bates, M.A., Brown, G., Bruce, B.C., Cook, C.C., Gould, L., Klovstad, J.W., Makhoul, J.I., Nash-Webber, B.L., Schwartz, R.M., Wolf, J.J. and Zue, V.W. (1975d) "Speech Understanding Systems, Quarterly Technical Progress Report No. 4, 30 October 1974 to 29 October 1975," Report No. 3188, Bolt Beranek and Newman Inc., Cambridge, Mass.
- [15] Woods, W.A., Bates, M.A., Brown, G., Bruce, B.C., Cook, C.C., Gould, L., Klovstad, J.W., Makhoul, J.I., Nash-Webber, B.L., Schwartz, R.M., Wolf, J.J. and Zue, V.W. (1976a) "Speech Understanding Systems, Quarterly Technical Progress Report No. 5, 30 October 1975 to 31 January 1976," Report No. 3240, Bolt Beranek and Newman Inc., Cambridge, Mass.
- [16] Woods, W.A., Bates, M.A., Brown, G., Bruce, B.C., Cook, C.C., Klovstad, J.W., Nash-Webber, B.L., Schwartz, R.M., Wolf, J.J. and Zue, V.W. (1976b) "Speech Understanding Systems, Quarterly Technical Progress Report No. 6, 1 February 1976 to 30 April 1976," Report No. 3303, Bolt Beranek and Newman Inc., Cambridge, Mass.
- [17] Woods, W.A., Bates, M.A., Brown, G., Bruce, B.C., Cook, C.C., Klovstad, J.W., Schwartz, R.M. and Wolf, J.J. (1976c) "Speech Understanding Systems, Quarterly Technical Progress Report No. 7, 1 May 1976 to 31 July 1976," Report No. 3359, Bolt Beranek and Newman Inc., Cambridge, Mass.
- [18] Woods, W.A., Bates, M.A., Brown, G., Bruce, B.C., Klovstad, J.W. and Nash-Webber, B.L. (1976) "Uses of Higher Level Knowledge in a Speech Understanding System: A Progress Report." Proc. IEEE-ASSP Meeting, Philadelphia, Pa.

[19]Woods, W.A., Kaplan, R.M. and Nash-Webber, B. (1972) "The Lunar Sciences Natural Language Information System: Final Report," BBN Report No. 2378, Bolt Beranek and Newman Inc., Cambridge, Mass.

[20]Woods, W.A., Schwartz, R.M., Cook, C.C., Klatt, D.H., Wolf, J.J., Bates, L.A., Nash-Webber, B.L., Bruce, B.C. and Makhoul, J.I. (1975b) "Speech Understanding Systems, Quarterly Technical Progress Report No. 2, 1 February 1975 to 1 May 1975," Report No. 3080, Bolt Beranek and Newman Inc., Cambridge, Mass.

[21]Woods, W.A., Schwartz, R.M., Cook, C.C., Klovstad, J.W., Bates, L.A., Nash-Webber, B.L., Bruce, B.C. and Makhoul, J.I. (1975c) "Speech Understanding Systems, Quarterly Technical Progress Report No. 3, 1 May 1975 to 1 August 1975," Report No. 3115, Bolt Beranek and Newman Inc., Cambridge, Mass.

[22]Woods, W.A., Schwartz, R.M., Klovstad, J.W., Cook, C.C., Wolf, J.J., Bates, M.A., Nash-Webber, B.L., Bruce, B.C. and Zue, V.W. (1975a) "Speech Understanding Systems, Quarterly Technical Progress Report No. 1, 1 November 1974 to 1 February 1975," Report No. 3018, Bolt Beranek and Newman Inc., Cambridge, Mass.

[23]Woods, W.A., Bates, M.A., Colarusso, J., Cook, C.C., Gould, L., Grabel, D., Makhoul, J.I., Nash-Webber, B.L., Schwartz, R.M. and Wolf, J.J. (1974) "Speech Understanding Research At BBN: Final Report, October 1970 to December 1974," Report No. 2976, Bolt Beranek and Newman Inc., Cambridge, Mass.

I. Appendices

1.1 Data Base Structures

This section presents a few examples from the nearly 2500 nodes in TRAVELNET. (See also Figures 14, 15, 16, 18, 19, 21 and 22 in the main text.) Generally a type node will be shown together with one or more instances of that type. The first node is that for DB/BUDGET.

728 - DB/BUDGET

CREATOR	CHIP
LINKS	(EGO) (BUDGET/OF) (CREATE/TIME) (DB/CREATOR) (MODALITY) (ITEMS) (PREVIOUS/BUDGET) (SUBSEQUENT/BUDGET) (SUB/BUDGETS) (SUPER/BUDGET) (MONEY/ALLOCATED) (MISC/POT)
INSTANCES	[BUDGET FOR SPEECH TRIPS 74-75 581] [BUDGET OF RE-IMBURSED SPEECH TRIPS 874] [BUDGET FOR SPEECH TRIPS 75-76 981] [SPEECH COMPRESSION TRAVEL BUDGET 74-75 1260] [ROBOT TRAVEL BUDGET 74-75 1269] [SPEECH TRANSCRIPTION TRAVEL BUDGET 75-76 1278] [SPEECH COMPRESSION TRAVEL BUDGET 75-76 1323]
VALUE/CLASS/OF	(ITEM/OF)
ENGLISH/NAME	(BUDGET)
SYNTAX/CLASS	(BUDGET)
ISA	(ACCOUNTING/CONCEPT)
G-FRAMES	1410 1411 1412
ACTION	(HAS)
LAST/MENTIONED	[BUDGET FOR SPEECH TRIPS 75-76 981]

Next are two instances of DB/BUDGET:

581 - (BUDGET FOR SPEECH TRIPS 74-75)

CREATOR	GREG
YEAR	1975
INSTANCE/OF	(DB/BUDGET)
DB/CREATOR	[GREG HARRIS 752]
ITEMS	1254 1258 1261 1265 1292
CREATE/TIME	713
BUDGET/OF	1107
MONEY/ALLOCATED	1287
MISC/POT	1292

981 - (BUDGET FOR SPEECH TRIPS 75-76)

CREATOR	LAURA
YEAR	1976
INSTANCE/OF	(DB/BUDGET)
DB/CREATOR	[JERRY WOLF 280]
CREATE/TIME	957
ITEMS	982 983 984 985 986 1003 1006 1009 1012 1019 1022 1043 1044 1045 1202 1207
BUDGET/OF	1205
MONEY/ALLOCATED	1288
MISC/POT	1207
LAST/MENTIONED/OF	(DB/BUDGET)

A DB/BUDGET is composed of a number of BUDGET/ITEMS. The next node is the link that connects a DB/BUDGET to its BUDGET/ITEMS.

722 - ITEMS

TYPE	LINKNAME
CREATOR	CHIP
MULTIPLE	T
*REL	(ITEM/OF)
LINK/OF	(DB/EXPEDITION) (DB/BUDGET)
INSTANCE/OF	(LINKNAME)

Next is the type node for BUDGET/ITEM, followed by a particular BUDGET/ITEM, 982.

1203 - BUDGET/ITEM

CREATOR	LAURA
INSTANCES	982 983 984 985 986 1003 1006 1009 1012 1019 1022 1043 1044 1045 1206 1207 1254 1258 1261 1265 1289 1290 1292 1294 1325
LINKS	(EGO) (ITEM/OF) DB/PURPOSE) (MONEY/REMAINING) (COVERS) (MONEY/ALLOCATED) (PLANS) (MONEY/SPENT) (MISC/POT/OF)
ENGLISH/NAME/OF	(COVERED/BY)
SYNTAX/CLASS	(BUDGET-ITEM)
ENGLISH/NAME	(BUDGET-ITEM)
ISA	(ACCOUNTING/CONCEPT)
G-FRAMES	1403 1404 1405 1406 1407 1408
LAST/MENTIONED	982

982

DB/PURPOSE	3 TO S.F. - ASA MEETING
CREATOR	LAURA
DB/CREATOR	[CHIP BRUCE 276]
CREATE/TIME	957
ITEM/OF	(BUDGET FOR SPEECH TRIPS 75-76 981)
INSTANCE/OF	(BUDGET/ITEM)
COVERS	1172 1351
PLANS	1225
MONEY/ALLOCATED	1291
LAST/MENTIONED/OF	(BUDGET/ITEM)

Both BUDGET/ITEMS and DB/BUDGETS have resources allocated to them. In the travel budget management domain the resource is money. The amount of money allocated is represented by an ALLOCATION node.

1248 - ALLOCATION

CREATOR LAURA

LINKS (UNITS) MONEY/REMAINING) (CURRENT/ALLOCATION)
(INITIAL/ALLOCATION) (ALLOCATION/OF)
(MONEY/SPENT)

INSTANCES 979 987 1202 1253 1255 1257 1259
1262 1264 1266 1270 1273 1280 1287
1288 1291 1293 1295 1296 1297 1298
1299 1300 1301 1302 1303 1304 1305
1306 1307

ISA (ACCOUNTING/CONCEPT)
INSTANCE/OF (ENGLISH/WORD)
G-FRAMES 1402

1291

CREATOR LAURA
INITIAL/ALLOCATION 1890
MONEY/SPENT 847.14
CURRENT/ALLOCATION 847.14
MONEY/REMAINING 0

INSTANCE/OF (ALLOCATION)
UNITS (DOLLARS)
ALLOCATION/OF 982

A DB/BUDGET is associated with a DB/CONTRACT. The DB/BUDGET represents the resources allocated and used, while the DB/CONTRACT contains information such as the PROJECT, SPONSOR, and TIME. Below are the type node for DB/CONTRACT, an instance of DB/CONTRACT, and SPONSOR, one of the links used by DB/CONTRACT.

217 - DB/CONTRACT

CREATOR CHIP

LINKS (GROUP) (BUDGET) (TIME) (ACCOUNT)
(PROJECT) (SPONSOR) (TRAVEL/MONEY)

INSTANCES 839 1107 1112 1116 1205 1277

ENGLISH/NAME (CONTRACT)
SYNTAX/CLASS (CONTRACT)
INSTANCE/OF (ENGLISH/WORD)
G-FRAMES 1416 1417 1418
ACTION (HAS)
LAST/MENTIONED 839

839

CREATOR CHIP
ACCOUNT 11321
YEAR 1976
TRAVEL/MONEY 0

INSTANCE/OF (DB/CONTRACT)
SPONSOR (ARPA)
TIME 1272
BUDGET [SPEECH COMPRESSION TRAVEL BUDGET 75-76 1323]
GROUP [SPEECH-COMPRESSION GROUP 1334]
MONITORED/BY (DSS-WASHINGTON)
PROJECT [SPEECH COMPRESSION 2372]
LAST/MENTIONED/OF (DB/CONTRACT)

542 - SPONSOR

TYPE	LINKNAME
CREATOR	CHIP
LINK/OF	(DB/CONTRACT) (DB/CONFERENCE)
*REL	(SPONSOR/OF)
METHODS	1190
VALUE/CLASS	(DB/SPONSOR)
SYNTAX/CLASS/OF	(DB/SPONSOR)
INSTANCE/OF	(LINKNAME)

BUDGET/ITEMs have ALLOCATIONS of resources, plans for using those resources (in this domain, TRAVEL/PLANS), and they cover various uses of those resources. In the TBM domain the only use of the resources is for trips. The type node for trips (DB/TRIP) is shown below, followed by an instance of a trip.

538 - DB/TRIP

CREATOR	CHIP
TYPE	LINKNAME
BUILDFN	BUILD-DB/TRIP
LINKS	(TIME) (COST) (ACCOUNT) (DESTINATION) (LEGS) (STARTING/POINT) (TRAVELER) (TRIP#) (CREATE/TIME) (DB/CREATOR) (TO/ATTEND) (DB/PURPOSE) (COVERED/BY)
INSTANCES	582 594 602 616 625 636 646 656 662 669 698 744 766 774 781 811 819 829 843 856 867 1127 1133 1161 1172 1351
METHODS	[Follow back pointers from time point to db/trip 1025]
VALUE/CLASS/OF	(LEG/OF) (ATTENDED/BY)
ENGLISH/NAME	(TRIP)
PROMPTLINKS	(COST) (ACCOUNT) (DESTINATION) (TRAVELER) (TRIP) (TO/ATTEND) (TRAVELERS) (COVERED/BY)
ISA	(ACCOUNTING/CONCEPT)
INSTANCE/OF	(LINKNAME)
G-FRAMES	1425 1426 1427 1428 1429 1430
ACTION	[IS GOING 2476]
LAST/MENTIONED	867

582

CREATOR	GREG
TRIP#	5914
ACCOUNT	230051
INSTANCE/OF	(DB/TRIP)
DB/CREATOR	[BILL WOODS 268]
TRAVELER	[JERRY WOLF 280]
LEGS	583 584
CREATE/TIME	592
COST	907 908
TIME	1053
COVERED/BY	1206
MODALITY	(TAKE)

Two of the links used in describing a DB/TRIP are TRAVELER and #TRAVELERS (i.e. number of travelers).

474 - TRAVELER

TYPE	LINKNAME
CREATOR	CHIP
ALIAS	TRAVELLER
*REL	(TRAVELER/OF)
LINK/OF	(DB/TRIP)
VALUE/CLASS	(PERSON)
METHODS	[Return number of travelers if traveler not specified 1047]
PROMPTLINK/OF	(DB/TRIP)
INSTANCE/OF	(LINKNAME)

883 - #TRAVELERS

CREATOR	LAURA
TYPE	LINKNAME
ALIAS	#TRAVELLERS
METHODS	[#TRAVELERS IS 1 IF THE TRAVELER IS SPECIFIED 1017]
PROMPTLINK/OF	(DB/TRIP)
LINK/OF	(TRAVEL/PLAN)
INSTANCE/OF	(LINKNAME)

The VALUE/CLASS (the type of nodes which can fill a slot) for TRAVELER is PERSON. Next are the type node for PERSON (abbreviated) followed by two instances of PERSON.

110 - PERSON

CREATOR	BONNIE
ALIAS	PEOPLE DB/PERSON
REFFN	FIXNAME
AGT/OF	[ATTENDING A MEETING 81] [CONCEPT OF GO 445] [CONCEPT OF TAKING A TRIP 451] [CONCEPT OF GO, AS TO ATTEND 453] [CONCEPT OF A TRIP 729] [CONCEPT OF ARRANGING A TRIP 736]
BENE/OF	[CONCEPT OF AVAILABILITY 106]
CAN/HAVE	[SENSE1 OF SCHEDULE - NOUN 230]
COMPONENT/OF	[GROUPS OF INDIVIDUALS 120]
INSTANCES	[LYN BATES 38] [LYNN COSELL 41] [JOHN MAKHOUL 43] [JOHN COLARUSSO 46] [RICH SCHWARTZ 261] [JACK KLOVSTAD 265] [BILL WOODS 268] [BONNIE NASH-WEBBER 270] [CRAIG COOK 272] [CHIP BRUCE 276] [DENNIS KLATT 278] [JERRY WOLF 280] [LINDA AMSDEN 282] [PAUL ROVNER 284] [BERT SUTHERLAND 286] [EQUIVCLASS OF I 289] [EQUIVCLASS OF WE 291] (SOMEONE) (ANYONE) [EQUIVCLASS OF HE 313] [EQUIVCLASS OF SHE 316] (WHO) [LAURA GOULD 501] [RAY NICKERSON 562] [THEY, AS PEOPLE 699] [GREG HARRIS 752] [MARYANN ROURKE 801] [JOE BECKER 840] [BILL MERRIAM 841]

VALUE/CLASS/OF	(TRAVELER) (DB/CREATOR)
OBJ/OF	[CONCEPT OF A LIST OF THINGS 737]
LINKS	(EGO) (FIRSTNAME) (LASTNAME) (MEMBER/OF) (GENDER)
SYNTAX/CLASS	(NAME)
ISA	(ANIMATE/OBJECT)
INSTANCE/OF	(ENGLISH/WORD)
G-FRAMES	1434
LAST/MENTIONED	[CHIP BRUCE 276]

38 - (LYN BATES)

CREATOR	BONNIE
FIRSTNAME	(LYN) (MADELEINE)
INSTANCE/OF	(PERSON)
LASTNAME	(BATES)
MEMBER/OF	[EQUIVCLASS OF SPEECH GROUP 55]
GENDER	(FEMALE)

841 - (BILL MERRIAM)

CREATOR	LAURA
INSTANCE/OF	(PERSON)
FIRSTNAME	(BILL)
LASTNAME	(MERRIAM)
GENDER	(MALE)
DB/CREATOR/OF	843 845 846 848 849 851 852 853 854
TRAVELER/OF	843
MEMBER/OF	[ROBOT GROUP 1333]

Much of the information about a DB/TRIP is associated with its "legs." Every DB/TRIP has at least two legs, one from the starting point to the destination and one to return. DB/TRIPs with intermediate stops have an additional LEG/OF/TRIP for each stop. Below are the type node for LEG/OF/TRIP and two instances of LEG/OF/TRIP.

714 - LEG/OF/TRIP

CREATOR	CHIP
LINKS	(TIME) (DESTINATION) (LEG/OF) (MODE/OF/TRANSPORT) (STARTING/POINT) (LEG) (CREATE/TIME) (DB/CREATOR) (DB/PURPOSE)
INSTANCES	583 584 595 601 607 610 621 623 629 631 642 644 652 654 659 661 665 666 673 675 705 709 759 762 764 770 772 777 779 789 792 795 797 815 817 821 824 826 833 835 846 849 852 854 860 862 871 873 1122 1128 1138 1141 1162 1165 1168 1208 1211 1353 1355
VALUE/CLASS/OF	(LEGS)
ISA	(ACCOUNTING/CONCEPT)

583

CREATOR	GREG
DB/PURPOSE	to discuss possible contract with Dr. William Rook of Life Sciences Research
LEG	1
INSTANCE/OF	(LEG/OF/TRIP)
DB/CREATOR	[BILL WOODS 268]
STARTING/POINT	[BOSTON MASSACHUSETTS 154]
DESTINATION	[WASHINGTON D.C. 294]
LEG/OF	582
TIME	586

873

CREATOR	LAURA
LEG	2
INSTANCE/OF	(LEG/OF/TRIP)
DB/CREATOR	[BILL WOODS 268]
LEG/OF	867
DESTINATION	[BOSTON MASSACHUSETTS 154]
STARTING/POINT	[SANTA BARBARA CALIFORNIA 162]
TIME	872

Next are two links used for a LEG/OF/TRIP. Note that for DESTINATION there is a METHOD (node 889) that allows one to refer to the DESTINATION of a DB/TRIP even though it is actually stored with the LEG/OF/TRIP. Similarly, there is a METHOD (node 895) that does the same for TIME.

455 - DESTINATION

TYPE	LINKNAME
CREATOR	CHIP
ADDFN	ADD-DESTINATION
IMPLIESFN	LOCATEDIN?
REMOVEFN	DELETE-EST-COST
*REL	(DESTINATION/OF)
LINK/OF	{DB/TRIP} (LEG/OF/TRIP) (TRAVEL/PLAN)
METHODS	{Get the destinations of a trip 889} {Finds the MEMBERS list for the CITY/PAIR corresponding to a fare. 1050}
VALUE/CLASS	(CITY)
PROMPTLINK/OF	{DB/TRIP}
INSTANCE/OF	(LINKNAME)

101 - TIME

IMPLIESFN	TENSE?
CREATOR	BONNIE
TYPE	LINKNAME
REMOVEFN	DELETE-EST-COST
ARG/TO	[AMOUNT OF TIME 105]
ISA	[A SPAN OF TIME 100] [MASS TERMS 239]
*REL	(TIME/OF)
LINK/OF	{DB/CONTRACT} {DB/TRIP} {DB/CONFERENCE}
METHODS	{LEG/OF/TRIP} (TRAVEL/PLAN) (DISCOURSE/STATE) {Get the time of a trip 895} [The time of a trip to a conference is assumed to be the time of the conference 599]
VALUE/CLASS	(TIME/PERIOD)
ENGLISH/NAME/OF	{TIME/PERIOD}
INSTANCE/OF	(LINKNAME)

TIME/PERIODs are the top level structures for representation of times see [Bates and Bruce, 1975]. They are composed of TIME/POINTS and a LENGTH/OF/TIME. A time period may be completely or partially specified. For example, one might know only the length of a trip and not its starting time; or one might know when it starts but not when it ends (or its length). The inference routines (see Section E.5) are able to process such information in various, incomplete forms and produce results at the maximum possible information level. The first item shown here is the type node for TIME/PERIOD (abbreviated):

534 - TIME/PERIOD

CREATOR	CHIP
REFFN	IDENTITY
ISA	(TIME/STRUCTURE)
LINKS	(BEGIN/TIME) (DURATION) (END/TIME)
	(TIME/OF) (CREATE/TIME) (DB/CREATOR)
INSTANCES	503 504 546 578 586 588 598 600
	606 620 622 628 630 641 643 651
	653 658 664 667 672 708 758 761
	763 769 771 776
	:
	:
	:
VALUE/CLASS/OF	(TIME) (DURATION/OF)
SYNTAX/CLASS	(DATE)
ENGLISH/NAME	(TIME)
G-FRAMES	1447 1448 1449 1450 1451 1452
LAST/MENTIONED	1038

Next are two instances of TIME/PERIODs:

503

CREATOR	CHIP
INSTANCE/OF	(TIME/PERIOD)
DB/CREATOR	[CHIP BRUCE 276]
CREATE/TIME	957
BEGIN/TIME	596
END/TIME	597
TIME/OF	594 1251 1252

1115

CREATOR	LAURA
INSTANCE/OF	(TIME/PERIOD)
DB/CREATOR	[CHIP BRUCE 276]
CREATE/TIME	957
BEGIN/TIME	1113
END/TIME	1114
TIME/OF	1116

Next is one of the links used for a given TIME/PERIOD:

275 - BEGIN/TIME

TYPE	LINKNAME
CREATOR	CHIP
ADDFN	ADD-TIME
IMPLIESFN	EQUALS?
*REL	(BEGIN/TIME/OF)
LINK/OF	(TIME/PERIOD)
VALUE/CLASS	(TIME/POINT)
METHODS	1193 [Get the beginning time for a trip 893]
INSTANCE/OF	(LINKNAME)

A TIME/POINT can be specified at any level of detail from "a Tuesday in June" to "at 3:15 p.m. on August 2nd, 1975". In the travel budget management data base times are normally kept only to the resolution of days. TIME/POINT's are linked together in a lattice by the PRECEDES relation. First is the type node for TIME/POINT (abbreviated):

535 - TIME/POINT

FINDFN	[NLAMBDA ALIST (TIMEBUILD (CDR ALIST]
CREATOR	CHIP
REFFN	STRUCTURE-TIME
BUILDFN	BUILD-TIME/POINT
ISA	(TIME/STRUCTURE)
VALUE/CLASS/OF	(BEGIN/TIME) (END/TIME) (PRECEDED/BY)
LINKS	(PRECEDES) (CREATE/TIME)
	(YEAR) (PRECEDED/BY) (PRECEDES)
	(DAY/OF/MONTH) (DAY/OF/WEEK) (HOUR)
	(MINUTE) (MONTH)
INSTANCES	419 461 536 576 577 585 592 593
	596 597 603 604 618 619 624 626
	627 632 639 640 645 647 648 655
	:
	:
	:
SYNTAX/CLASS	(DATE)
G-FRAMES	1453 1454 1455 1456 1457 1458 1459
LAST/MENTIONED	1110

Next are two instances of TIME/POINTS:

419

CREATOR	GREG
DAY/OF/MONTH	1
MONTH	4
YEAR	1975
INSTANCE/OF	(TIME/POINT)
CREATE/TIME/OF	602
PRECEDES	603
PRECEDED/BY	855

1114

CREATOR	LAURA
DAY/OF/MONTH	21
MONTH	11
YEAR	1975
INSTANCE/OF	(TIME/POINT)
END/TIME/OF	1115
PRECEDES	1176
PRECEDED/By	1213

A LENGTH/OF/TIME is another component of a TIME/PERIOD. Like TIME/POINTS, LENGTH/OF/TIMES can be specified at various levels of detail. Below are the type node for LENGTH/OF/TIME and two instances.

533 - LENGTH/OF/TIME

CREATOR	CHIP
REFFN	STRUCTURE-DURATION
ISA	(TIME/STRUCTURE)
VALUE/CLASS/OF	(DURATION)
LINKS	(DURATION/OF) (YEARS) (MONTHS)
	(WEEKS) (DAYS) (HOURS) (MINUTES)
INSTANCES	989 991 992 994 995 1004
G-FRAMES	1432 1433

989

CREATOR	LAURA
DAYS	4
INSTANCE/OF	(LENGTH/OF/TIME)
DURATION/OF	978 990 997 1179 1183 1183 1210
	1210 1210 1210 1210 1210 1210 1210
	1210 1210 1210 1210

1004

CREATOR	LAURA
DAYS	3
INSTANCE/OF	(LENGTH/OF/TIME)
DURATION/OF	504 1007 1010 1013 1020

The next set of nodes exemplifies the storage of fare information. First is the type node for DB/FARE, then two instances of DB/FARE, and finally, one of the links used in a DB/FARE.

485 - DB/FARE

BUILDFN	COMPLETE:
LINKS	(TYPE) (MODE/OF/TRANSPORT) (CREATE/TIME)
INSTANCES	(DB/CREATOR) (VALUE) (FARE/OF)
	159 477 798 949 950 951 952 953
	954 955 1054 1069 1070 1072 1073
	1075 1077 1078 1079 1081 1083 1085
	1087 1091 1093 1095 1096 1097 1098
	1099 1100 1135 1155 1157 1159 1174
	1335 1360
ENGLISH/NAME	(FARE)
PROMPTLINKS	(TYPE) (MODE/OF/TRANSPORT) (VALUE)
	(FARE/OF)
ISA	(ACCOUNTING/CONCEPT) (DB/EXPENSE)
G-FRAMES	1421 1422
LAST/MENTIONED	1095

159

CREATOR	LAURA
VALUE	50
INSTANCE/OF	(DB/FARE)
DB/CREATOR	[LAURA GOULD 501]
CREATE/TIME	957
MODE/OF/TRANSPORT	(TRAIN)
FARE/OF	1089
UNITS	(DOLLARS)

1100

CREATOR	LAURA
VALUE	39
INSTANCE/OF	(DB/FARE)
DB/CREATOR	[LAURA GOULD 501]
CREATE/TIME	957
MODE/OF/TRANSPORT	(AIR)
FARE/OF	880
UNITS	(DOLLARS)

465 - MODE/OF/TRANSPORT

TYPE	LINKNAME
CREATOR	CHIP
DEFAULT/VALUE	67
*REL	(TRANSPORT/OF)
LINK/OF	(DB/FARE) (LEG/OF/TRIP)
METHODS	[get the means of transportation for a trip 891]
	[Unique low-ranked METHOD assumes defaults 802]
PROMPTLINK/OF	(DB/FARE)
INSTANCE/OF	(LINKNAME)

Fares are defined for pairs of cities (CITY/PAIR). Because (1) our data base for fares is sparse, and (2) we want to associate mileage (and potentially other information) with a CITY/PAIR, we have a separate node for each pair. This is in contrast to a matrix format for storage. Note that only a very small number of potential CITY/PAIRs have been put in the data base.

543 - CITY/PAIR

CREATOR	CHIP
LINKS	(FARE) (MEMBERS) (MILEAGE)
INSTANCES	552 553 554 555 556 718 877 878 879 880 1055 1071 1074 1076 1080 1082 1084 1086 1088 1089 1090 1092 1094 1124 1125 1134 1156 1158 1160 1175
VALUE/CLASS/OF	(FARE/OF)
ISA	(ABSTRACT/ENTITY)

552

MILEAGE	3265
CREATOR	CHIP
MEMBERS	[LONDON ENGLAND 150] [BOSTON MASSACHUSETTS 154]
INSTANCE/OF	(CITY/PAIR)
FARE	477

An abbreviated listing of the node for CITY is shown next, followed by three instances of CITY.

293 - CITY

CREATOR	BONNIE
PROPOSECAT	CITY
INSTANCES	[LONDON ENGLAND 150] [AMHERST MASSACHUSETTS 152] [BOSTON MASSACHUSETTS 154] [EQUIV/CLASS OF U.S.A. 155] [PITTSBURGH PENNSYLVANIA 160] [SANTA BARBARA CALIFORNIA 162] [STOCKHOLM SWEDEN 163] [WASHINGTON D.C. 294] [NEW YORK CITY NEW YORK 296] [TBILISI RUSSIA 307] [OTTAWA CANADA 548] [ST LOUIS MISSOURI 551] [AUSTIN TEXAS 609] [SAN DIEGO CALIFORNIA 617] : :
ISA	[SPATIAL LOCATION 719]
OBJ/OF	[CONCEPT OF A LIST OF THINGS 737]
LINKS	{LOCATION}
VALUE/CLASS/OF	{LOCATION} (MEMBERS) (DESTINATION) (STARTING/POINT)
SYNTAX/CLASS/OF	{CITY} (LOCATION)
SYNTAX/CLASS	{CITY}
INSTANCE/OF	(ENGLISH/WORD)
G-FRAMES	1409

150 - (LONDON ENGLAND)

CREATOR	BONNIE
INSTANCE/OF	(CITY)
LOCATION	(ENGLAND)
MEMBER/OF	552
CITY/NAME	(LONDON)

1037 - (HOUSTON TEXAS)

CREATOR LAURA
INSTANCE/OF (CITY)
LOCATION (TEXAS)
LOCATION/OF 1041
DESTINATION/OF 1239
MEMBER/OF 1055
CITY/NAME (HOUSTON)

154 - (BOSTON MASSACHUSETTS)

CREATOR BONNIE
INSTANCE/OF (CITY)
LOCATION (MASSACHUSETTS)
MEMBER/OF 552 553 554 555 556 718 877 878
879 880 1054 1055 1071 1074 1076
1080 1082 1086 1090 1124 1125 1134
1156 1158 1175
LOCATION/OF [EQUIVCLASS OF SPEECH GROUP 55]
462 [ROBOT GROUP 1333] [SPEECH-COMPRESSION GROUP
1334] [Current discourse state 1472]
STARTING/POINT/OF 583 595 607 621 629 642 652 659
665 673 705 759 770 777 789 815
821 833 846 860 871 1122 1138 1162
1208 1353
DESTINATION/OF 584 601 610 623 631 644 654 661
666 675 709 764 772 779 797 817
826 835 854 862 873 1128 1141 1168
1211 1355
CITY/NAME (BOSTON)

The next node is the type node for conferences. Following that
are two instances of DB/CONFERENCE.

637 - DB/CONFERENCE

CREATOR CHIP
TYPE LINKNAME
ALIAS DB/MEETING
LINKS (TIME) (LOCATION) (SPONSOR) (CREATE/TIME)
(DB/CREATOR) (ATTENDED/BY)
INSTANCES 462 579 888 961 965 972 976 1029
1034 1041 1251 1256
METHODS [Follow back pointers from time point
to db/conference 956]
VALUE/CLASS/OF (TO/ATTEND) (REGISTRATION/FEE/OF)
ENGLISH/NAME / CONFERENCE)
SYNTAX/CLASS (MEETING)
INSTANCE/OF (LINKNAME)
G-FRAMES 1413 1414 1415
ACTION [WILL BE HELD 2471]
LAST/MENTIONED 462

462

CREATOR CHIP
INSTANCE/OF (DB/CONFERENCE)
SPONSOR (ACL)
LOCATION [BOSTON MASSACHUSETTS 154]
TIME 546
DB/CREATOR [CHIP BRUCE 276]
REGISTRATION/FEE 901
LAST/MENTIONED/OF (DB/CONFERENCE)

1041

```
CREATOR          LAURA
INSTANCE/OF      (DB/CONFERENCE)
DB/CREATOR       [JERRY WOLF 280]
CREATE/TIME      957
SPONSOR          (ACM)
LOCATION          [HOUSTON TEXAS 1037]
TIME            1042
ATTENDED/BY     1239
```

A DB/CONFERENCE may have a SPONSOR (e.g. ACL or ACM). Below are the type node for CONFERENCE/SPONSOR and ACL, an instance of CONFERENCE/SPONSOR.

1118 - CONFERENCE/SPONSOR

```
CREATOR          CHIP
INSTANCES        (ACL) (ASA) (IEEE) (IJCAI-COMMITTEE)
                  (ASSP) (ICCL) (A.I.M.) (ACM) (NCC-COMMITTEE)
                  (COMPCON-COMMITTEE)
LINKS            (SPONSOR/OF)
ISA              (DB/SPONSOR)
INSTANCE/OF      (ENGLISH/WORD)
```

12 - ACL

```
CREATOR          BONNIE
EQUIV/TO         [EQUIVCLASS OF ACL 11]
SPONSOR/OF       462
INSTANCE/OF      (CONFERENCE/SPONSOR) (ENGLISH/WORD)
```

Finally, we have the type node for DB/FEE followed by an instance of DB/FEE, the registration fee for the ACL conference.

899 - DB/FEE

```
CREATOR          CHIP
BUILDFN          COM.LETE:
LINKS            (TYPE) (CREATE/TIME) (DB/CREATOR)
                  (REGISTRATION/FEE/OF) (VALUE) (UNITS)
INSTANCES        901 902 1361
ENGLISH/NAME     (FEE)
PROMPTLINKS      (TYPE) (REGISTRATION/FEE/OF) (VALUE)
ISA              (ACCOUNTING/CONCEPT) (DB/EXPENSE)
G-FRAMES         1423 1424
LAST/MENTIONED   901
```

901

```
CREATOR          CHIP
VALUE            20
INSTANCE/OF      (DB/FEE)
DB/CREATOR       [CHIP BRUCE 276]
REGISTRATION/FEE/OF 462
CREATE/TIME      95
UNITS            (DOLLARS)
LAST/MENTIONED/OF (DB/FEE)
```

I.2 Example Parses and Interpretations

This section contains a number of example sentences. In each case there is first the sentence as it appears following action by SPEECHIFY (Section D.2.1), then the syntactic parse (Section D.2.2), the semantic interpretation (Section D.3), and the optimized interpretation (Section E.3). Further examples are shown in Volume 4, Appendix 2.

```

*****
*
* ENTER A TRIP FOR BONNIE NASH-WEBBER TO GRENOBLE *
*
*****

```

Found complete parse:

```

S IMP
  SUBJ NP PRO YOU
        FEATS NU SG
  AUX TNS PRESENT
        VOICE ACTIVE
  VP V ENTER
      OBJ NP DET ART A
            N TRIP
            PP PREP FOR
              NPR NAME FIRST BONNIE
                    LAST NASH-WEBBER
            PP PREP TO
              NP NPP LOCATION CITY GRENOBLE
              FEATS NU SG

```

Interpretation:

```

(FCR: THE A0085 / (FINDQ: PERSON (LASTNAME NASH-WEBBER)
                    (FIRSTNAME BONNIE))
  : T ; (FOR: THE A0086 / (FINDQ: LOCATION (CITY GRENOBLE))
        : T ; (FOR: 1 A0087 / (BUILDQ: DB/TRIP
                                (DESTINATION A0086)
                                (TRAVELER A0085))
        : T ; T)))

```

Optimized interpretation.

```

(FOR: THE A0085 / (FIND: (INSTANCE/OF (QUOTE PERSON))
                        (LASTNAME (QUOTE NASH-WEBBER))
                        (FIRSTNAME (QUOTE BONNIE)))
  : T ; (FOR: THE A0086 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
                                (CITY (QUOTE GRENOBLE)))
        : T ; (FOR: 1 A0087 / (BUILD: DB/TRIP
                                (DESTINATION A0086)
                                (TRAVELER A0085))
        : T ; T)))

```

```

*****
*
* HOW MUCH DID WE SPEND IN DECEMBER *
*
*****

```

Found complete parse:

```

S Q
  SUBJ NP PRO WE
        FEATS NU PL
  AUX TNS PAST
        VOICE ACTIVE
  VP V SPEND
      OBJ NP DET QUANT HOWMUCH
            N ?
            FEATS NU MASS
  PP PREP IN
        TIME MONTH DECEMBER

```

Interpretation:

```

(FOR: ALL A0108 / (FINDQ: DB/BUDGET (TIME (MONTH DECEMBER))
                  (BUDGET/FOR SPEAKER))
: T ; (OUTPUT: (GET: A0108 COST)))

```

Optimized interpretation:

```

[FOR: ALL A0108 / (FIND: (INSTANCE/OF (QUOTE DB/BUDGET))
                        (TIME (QUOTE (MONTH DECEMBER)))
                        (BUDGET/FOR (QUOTE SPEAKER)))
: T ; (OUTPUT: (GET: A0108 (QUOTE COST)]

```

```

*****
*
* MY TRIP TO MEXICO COST-PAST EIGHT HUNDRED DOLLAR-S *
*
*****

```

Found complete parse:

```

S DCL
  SUBJ NP DET POSS MY
        N TRIP
        PP PREP TO
          NP NPR LOCATION COUNTRY MEXICO
        FEATS NU SG
  AUX TNS PAST
        VOICE ACTIVE
  VP V COST
    OBJ NP $ 800

```

Interpretation:

```

[FOR: THE A0129 / (FINDQ: LOCATION (COUNTRY MEXICO))
  : T ; (FOR: THE A0130 / (FINDQ: DB/TRIP (DESTINATION A0129)
                          (TIME (BEFORE NOW))
                          (TRAVELER SPEAKER))
  : T ; (PROGN (ADD: A0130 VALUE 800)
            (ADD: A0130 UNITS DOLLARS])

```

Optimized interpretation:

```

[FOR: THE A0129 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
                     (COUNTRY (QUOTE MEXICO)))
  : T ; (FOR: THE A0130 / (FIND: (INSTANCE/OF (QUOTE DB/TRIP))
                                (DESTINATION A0129)
                                (TIME (QUOTE PAST))
                                (TRAVELER (QUOTE SPEAKER)))
  : T ; (PROGN (ADD: A0130 (QUOTE VALUE)
                  (QUOTE 800))
            (ADD: A0130 (QUOTE UNITS)
                  (QUOTE DOLLARS)])

```

```
*****
*
* SHOW ME HER TRIP-S *
*
*****
```

Found complete parse:

```
S IMP
  SUBJ NP PRO YOU
        FEATS NU SG
  AUX TNS PRESENT
        VOICE ACTIVE
  VP V SHOW
    OBJ NP DET POSS HER
          N TRIP
          FEATS NU PL
```

Interpretation:

```
(FOR: THAT A0021 / (FINDQ: DB/PERSON (GENDER FEMALE))
  : T ; (FOR: EVERY A0022 / (FINDQ: DB/TRIP (TRAVELER A0021))
    : T ; (OUTPUT: A0022)))
```

Optimized interpretation:

```
(FOR: THAT A0021 / (FIND: (INSTANCE/OF (QUOTE DB/PERSON))
                        (GENDER (QUOTE FEMALE)))
  : T ; (FOR: EVERY A0022 / (FIND: (INSTANCE/OF (QUOTE DB/TRIP))
                                (TRAVELER A0021))
    : T ; (OUTPUT: A0022)))
```

```
*****
*
* HOW MANY TRIP-S HAS RICH TAKEN *
*
*****
```

Found complete parse:

```
S Q
SUBJ NPR NAME FIRST RICH
AUX TNS PAST
    VOICE ACTIVE
" P V TAKE
    OBJ NP DET QUANT HOW@MANY
        N TRIP
        FEATS NU PL
```

Interpretation:

```
[FOR: THE A0093 / (FINDQ: PERSON (FIRSTNAME RICH))
: T ; (FOR: THE A0092 / [SETOF: (FINDQ: DB/TRIP (TRAVELER A0093)
                                (TIME (BEFORE NOW]
: T ; (OUTPUT: (COUNT: A0092]
```

Optimized interpretation:

```
[FOR: THE A0093 / (FIND: (INSTANCE/OF (QUOTE PERSON))
                    (FIRSTNAME (QUOTE RICH)))
: T ; (FOR: THE A0092 / [SETOF: (FINDQ: DB/TRIP (TRAVELER A0093)
                                (TIME (BEFORE NOW]
: T ; (OUTPUT: (COUNT: A0092]
```

```
*****
*
* WHAT IS THE PLANE FARE TO OTTAWA *
*
*****
```

Found complete parse:

```
S Q
  SUBJ NP DET ART THE
        ADJ PLANE
        N FARE
        PP PREP TO
          NP NPR LOCATION CITY OTTAWA
        FEATS NU SG
  AUX TNS PRESENT
        VOICE ACTIVE
  VP V BE
    OBJ NP PRO WHAT
          FEATS NU SG/PL
```

Interpretation:

```
(FOR: THE A0064 / (FINDQ: LOCATION (CITY OTTAWA))
  : T ; (FOR: THE A0066 / (FINDQ: DB/FARE (DESTINATION A0064)
                          (STARTING/POINT BOSTON)
                          (MODE/OF/TRANSPORT PLANE)))
    : T ; (OUTPUT: A0066)))
```

Optimized interpretation:

```
(FOR: THE A0064 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
                      (CITY (QUOTE OTTAWA)))
  : T ; (FOR: THE A0066 / (FIND: (INSTANCE/OF (QUOTE DB/FARE))
                                (DESTINATION A0064)
                                (STARTING/POINT (QUOTE BOSTON))
                                (MODE/OF/TRANSPORT (QUOTE PLANE)))
    : T ; (OUTPUT: A0066)))
```

```

*****
*
* WHY DID JERRY GO TO UNIVAC *
*
*****

```

Found complete parse:

```

S Q
QADV WHY
S Q
SUBJ NPR NAME FIRST JERRY
AUX TNS PAST
    VOICE ACTIVE
VP V GO
    PP PREP TO
        NP N UNIVAC
            FEATS NU SG
ADV WHY

```

Interpretation:

```

[FOR: THE A0106 / (FINDQ: DB/CONFERENCE (LOCATION UNIVAC))
: T ; (FOR: THE A0105 / (FINDQ: PERSON (FIRSTNAME JERRY))
: T ; (FOR: ALL A0107 / (FINDQ: DB/TRIP
                                (TRAVELER A0105)
                                (TO/ATTEND A0106)
                                (TIME (BEFORE NOW)))
: T ; (OUTPUT: (GET: A0107 PURPOSE]

```

Optimized interpretation:

```

[FOR: THE A0106 / (FIND: (INSTANCE/OF (QUOTE DB/CONFERENCE))
                                (LOCATION (QUOTE UNIVAC)))
: T ; (FOR: THE A0105 / (FIND: (INSTANCE/OF (QUOTE PERSON))
                                (FIRSTNAME (QUOTE JERRY)))
: T ; (FOR: ALL A0107 / (FIND: (INSTANCE/OF
                                (QUOTE DB/TRIP))
                                (TRAVELER A0105)
                                (TO/ATTEND A0106)
                                (TIME (QUOTE PAST)))
: T ; (OUTPUT: (GET: A0107 (QUOTE PURPOSE]

```

```
*****
*
* WHAT IS THE REGISTRATION FEE FOR THE NEXT ACL CONFERENCE *
*
*****
```

Found complete parse:

```
S Q
  SUBJ NP DET ART THE
        ADJ N REGISTRATION
        N FEE
        PP PREP FOR
          NP DET ART THE
            ADJ NEXT
            ADJ NP NPR ACL
            N CONFERENCE
            FEATS NU SG
          FEATS NU SG
  AUX TNS PRESENT
    VOICE ACTIVE
  VP V BE
    OBJ NP PRO WHAT
      FEATS NU SG/PL
```

Interpretation:

```
(FOR: (ORDINAL NEXT)
  A0134 / (FINDQ: DB/CONFERENCE (SPONSOR ACL))
  : T ; (FOR: THE A0136 / (FINDQ: DB/FEE (FEE/OF A0134)
                          (TYPE REGISTRATION))
  : T ; (OUTPUT: A0136)))
```

Optimized interpretation:

```
(FOR: (ORDINAL NEXT)
  A0134 / (FIND: (INSTANCE/OF (QUOTE DB/CONFERENCE))
                (SPONSOR (QUOTE ACL)))
  : T ; (FOR: THE A0136 / (FIND: (INSTANCE/OF (QUOTE DB/FEE))
                                (FEE/OF A0134)
                                (TYPE (QUOTE REGISTRATION)))
  : T ; (OUTPUT: A0136)))
```

```
*****
*
* WHO WENT TO SANTA@BARBARA IN AUGUST *
*
*****
```

Found complete parse:

```
S Q
  SUBJ NP PRO WHO
        FEATS NU SG/PL
  AUX TNS PAST
        VOICE ACTIVE
  VP V GO
    PP PREP TO
      NP NPR LOCATION CITY SANTA@BARBARA
    PP PREP IN
      TIME MONTH AUGUST
```

Interpretation:

```
[FOR: THE A0102 / (FINDQ: LOCATION (CITY SANTA@BARBARA))
  : T ; (FOR: ALL A0103 / (FINDQ: DB/TRIP (DESTINATION A0102)
                          (TIME (MONTH AUGUST)))
  : T ; (OUTPUT: (GET: A0103 TRAVELER]
```

Optimized interpretation:

```
[FOR: THE A0102 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
                     (CITY (QUOTE SANTA@BARBARA)))
  : T ; (FOR: ALL A0103 / [FIND: (INSTANCE/OF (QUOTE DB/TRIP))
                              (DESTINATION A0102)
                              (TIME (QUOTE (MONTH AUGUST))
  : T ; (OUTPUT: (GET: A0103 (QUOTE TRAVELER]
```

```

*****
*
* WHEN IS THE NEXT ASA MEETING *
*
*****

```

Found complete parse:

```

S Q
QADV WHEN
S Q
  SUBJ NP DET ART THE
        ADJ NEXT
        ADJ NP NPR ASA
        N MEETING
        FEATS NU SG
  AUX TNS PRESENT
        VOICE ACTIVE
  VP V BE
  ADV THEN

```

Interpretation:

```

(FOR: (ORDINAL NEXT)
  A0067 / (FINDQ: DB/CONFERENCE (SPONSOR ASA)
            (TIME (AFTER NOW)))
: T ; (OUTPUT: (GET: A0067 TIME)))

```

Optimized interpretation:

```

[FOR: (ORDINAL NEXT)
  A0067 / (FIND: (INSTANCE/OF (QUOTE DB/CONFERENCE))
                 (SPONSOR (QUOTE ASA))
                 (TIME (QUOTE FUTURE)))
: T ; (OUTPUT: (GET: A0067 (QUOTE TIME)]

```

```
*****
*
* SHOW ME ALL THE TRIP-S TO EL@PASO *
*
*****
```

Found complete parse:

```
S IMP
  SUBJ NP PRO YOU
        FEATS NU SG
  AUX TNS PRESENT
        VOICE ACTIVE
  VP V SHOW
    OBJ NP DET QUANT ALL
          N PRO ONE
          PP PREP OF
            NP DET THE
              N TRIP
              PP PREP TO
                NP NPR LOCATION CITY EL@PASO
                FEATS NU PL
                FEATS NU PL
```

Interpretation:

```
(FOR: THE A0151 / (FINDQ: LOCATION (CITY EL@PASO))
 : T ; (FOR: ALL A0152 / (FINDQ: DB/TRIP (DESTINATION A0151))
 : T ; (CUTPUT: A0152)))
```

Optimized interpretation:

```
(FOR: THE A0151 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
                     (CITY (QUOTE EL@PASO)))
 : T ; (FOR: ALL A0152 / (FIND: (INSTANCE/OF (QUOTE DB/TRIP));
                           (DESTINATION A0151))
 : T ; (OUTPUT: A0152)))
```

```

*****
*
* WHAT TRIP-S REMAIN IN THE SPEECH BUDGET *
*
*****

```

Found complete parse:

```

S Q
  SUBJ NP DET WHAT
        N TRIP
        FEATS NU PL
  AUX TNS PRES
        VOICE ACTIVE
  VP V REMAIN
    PP PREP IN
      NP DET ART THE
        SPEECH
        N BUDGET
        FEATS NU SG

```

Interpretation:

```

(FOR: THE A0123 / (FINDQ: DB/BUDGET (PROJECT (SPEECH)))
: T ; (FOR: EVERY A0122 / (FINDQ: DB/TRIP (COVERED/BY A0123)
                           (TIME (AFTER NOW)))
: T ; (OUTPUT: A0122)))

```

Optimized interpretation:

```

(FOR: THE A0123 / [FIND: (INSTANCE/OF (QUOTE DB/BUDGET))
                       (PROJECT (QUOTE (SPEECH]
: T ; (FOR: EVERY A0122 / (FIND: (INSTANCE/OF (QUOTE DB/TRIP))
                              (COVERED/BY A0123)
                              (TIME (QUOTE FUTURE)))
: T ; (OUTPUT: A0122)))

```

```

*****
*                                     *
* SCHEDULE A TRIP BY TRAIN TO NEW@YORK *
*                                     *
*****

```

Found complete parse:

```

S IMP
  SUBJ NP PRO YOU
        FEATS NU SG
  AUX TNS PRESENT
        VOICE ACTIVE
  VP V SCHEDULE
    OBJ NP DET ART A
          N TRIP
          PP PREP BY
            NP N TRAIN
              FEATS NU SG
            PP PREP TO
              NP NPR LOCATION AMBIG NEW@YORK
              FEATS NU SG

```

Interpretation:

```

(FOR: THE A0072 / (FINDQ: LOCATION (AMBIG NEW@YORK))
  : T ; (FOR: 1 A0073 / (BUILDQ: DB/TRIP (MODE/OF/TRANSPORT TRAIN)
                        (DESTINATION A0072))
    : T ; T))

```

Optimized interpretation:

```

(FOR: THE A0072 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
                        (AMBIG (QUOTE NEW@YORK)))
  : T ; (FOR: 1 A0073 / (BUILD: DB/TRIP (MODE/OF/TRANSPORT
                        (QUOTE TRAIN))
                        (DESTINATION A0072))
    : T ; T))

```

✓

```
*****
*
* PLEASE SHOW ME JOHN MAKHOUL -S THREE TRIP-S TO PITTSBURGH *
*
*****
```

Found complete parse:

```
S IMP
SUBJ NP PRO YOU
      FEATS NU SG
AUX TNS PRESENT
      VOICE ACTIVE
VP V SHOW
  OBJ NP DET THE
        ADJ 3
        N TRIP
        PP PREP FOR
          NP NPR NAME FIRST JOHN
            LAST MAKHOUL
        PP PREP TO
          NP NPR LOCATION CITY PITTSBURGH
        FEATS NU PL
```

Interpretation:

```
[FOR: THE A0131 / (FINDQ: PERSON (LASTNAME MAKHOUL)
                      (FIRSTNAME JOHN))
: T ; (FOR: THE A0132 / (FINDQ: LOCATION (CITY PITTSBURGH))
: T ; (FOR: (THE 3)
          A0133 / (FINDQ: DB/TRIP (DESTINATION A0132)
                      (TRAVELER A0131))
: T ; (OUTPUT: A0133]
```

Optimized interpretation:

```
[FOR: THE A0131 / (FIND: (INSTANCE/OF (QUOTE PERSON))
                      (LASTNAME (QUOTE MAKHOUL))
                      (FIRSTNAME (QUOTE JOHN)))
: T ; (FOR: THE A0132 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
                              (CITY (QUOTE PITTSBURGH)))
: T ; (FOR: (THE 3)
          A0133 / (FIND: (INSTANCE/OF (QUOTE DB/TRIP))
                      (DESTINATION A0132)
                      (TRAVELER A0131))
: T ; (OUTPUT: A0133]
```

```
*****
*
* WHEN DID CRAIG GO TO UTAH *
*
*****
```

Found complete parse:

```
S Q
  QADV WHEN
S Q
  SUBJ NPR NAME FIRST CRAIG
  AUX TNS PAST
    VOICE ACTIVE
  VP V GO
    PP PREP TO
      NP NPR LOCATION STATE UTAH
  ADV THEN
```

Interpretation:

```
[FOR: THE A0062 / (FINDQ: LOCATION (STATE UTAH))
  : T ; (FOR: THE A0061 / (FINDQ: PERSON (FIRSTNAME CRAIG))
    : T ; (FOR: ALL A0063 / (FINDQ: DB/TRIP
      (TRAVELER A0061)
      (DESTINATION A0062)
      (TIME (BEFORE NOW)))
    : T ; (OUTPUT: (GET: A0063 TIME]
```

Optimized interpretation:

```
[FOR: THE A0062 / (FINDLOC (INSTANCE/OF (QUOTE LOCATION))
  (STATE (QUOTE UTAH)))
  : T ; (FOR: THE A0061 / (FIND: (INSTANCE/OF (QUOTE PERSON))
    (FIRSTNAME (QUOTE CRAIG)))
    : T ; (FOR: ALL A0063 / (FIND: (INSTANCE/OF
      (QUOTE DB/TRIP))
      (TRAVELER A0061)
      (DESTINATION A0062)
      (TIME (QUOTE PAST)))
    : T ; (OUTPUT: (GET: A0063 (QUOTE TIME]
```

1.3 Methods

This section contains the complete set of METHODS used for accessing data base items, for computation and for inference (see Section E.5).

125 - (FOLLOW TIME LINK; RETRY DURATION)

METHOD/RANK 20
ARGUMENT/PATHS (X NODE TIME DURATION)
FUNCTION IDENTITY
CREATOR GREG

METHOD/OF (DURATION)
INSTANCE/OF (METHOD)

587 - (Get the end points for a fare)

FUNCTION IDENTITY
APPLICABILITY/TEST DB/FARE?
ARGUMENT/PATHS (X NODE FARE/OF MEMBERS)
CREATOR CHIP
METHOD/RANK 25

INSTANCE/OF (METHOD)
METHOD/OF (END/POINTS)

599 - (The time of a trip to a conference is assumed to be the time of the conference)

FUNCTION [LAMBDA (X)
(APPLY* (FUNCTION BUILD:)
(QUOTE TIME/PERIOD)
(LIST (QUOTE TIME/OF)
NODE))
(ADD: LASTTIME/PERIOD (QUOTE BEGIN/TIME)
(GET: X (QUOTE BEGIN/TIME)
NIL
(QUOTE DONTASK)))
(ADD: LASTTIME/PERIOD (QUOTE END/TIME)
(GET: X (QUOTE END/TIME)
NIL
(QUOTE DONTASK))
ARGUMENT/PATHS (X NODE TO/ATTEND TIME)
APPLICABILITY/TEST [LAMBDA (NODE LINK VALUE)
(AND (INSTANCE/OF? NODE (QUOTE DB/TRIP))
(GET: NODE (QUOTE TO/ATTEND)
NIL T])
CREATOR CHIP
METHOD/RANK 40
INSTANCE/OF (METHOD)
METHOD/OF (TIME)

755 - (MEASURE LENGTH OF TIME/PERIOD: DURATION)

METHOD/RANK 10
ARGUMENT/PATHS (AFTER NODE END/TIME)(BEFORE NODE BEGIN/TIME)
FUNCTION DIFFERENCE/IN/TIME
APPLICABILITY/TEST HAS-TIME?
CREATOR GREG

INSTANCE/OF (METHOD)
METHOD/OF (DURATION)

802 - (Unique low-ranked METHOD assumes defaults)

CREATOR GREG
ARGUMENT/PATHS (X LINK DEFAULT/VALUE)
INSTANCE/OF (METHOD)
METHOD/OF (ACCUMULATE) (TYPE) (MODE/OF/TRANSPORT)
(STARTING/POINT) (HOUR) (MINUTE) (PER/DIEM)
(SECONDS) (YEARS) (MONTHS) (WEEKS)
(DAYS) (HOURS) (MINUTES)

875 - (INHERIT MODALITY FROM BUDGET)

CREATOR	GREG
FUNCTION	IDENTITY
ARGUMENT/PATHS	(X NODE ITEM/OF MODALITY)
APPLICABILITY/TEST	[LAMBDA (NODE LINK VALUE) (LINKFOL NODE (QUOTE ITEM/OF)]
METHOD/RANK	20
INSTANCE/OF	(METHOD)
METHOD/OF	(MODALITY)

876 - (COUNT THE LEGS OF THE TRIP)

APPLICABILITY/TEST	[LAMBDA (NODE LINK VALUE) (LINKFOL NODE (QUOTE LEGS)]
ARGUMENT/PATHS	(X NODE LEGS)
CREATOR	GREG
FUNCTION	COUNT
METHOD/RANK	20
INSTANCE/OF	(METHOD)
METHOD/OF	(NLEGS)

889 - (Get the destinations of a trip)

APPLICABILITY/TEST	TRIPWITHLEGS?
ARGUMENT/PATHS	(L NODE LEGS DESTINATION)
CREATOR	CHIP
METHOD/RANK	10
FUNCTION	BUTLAST
INSTANCE/OF	(METHOD)
METHOD/OF	(DESTINATION)

890 - (Get the purpose of a trip)

APPLICABILITY/TEST	TRIPWITHLEGS?
ARGUMENT/PATHS	(X NODE LEGS PURPOSE)
CREATOR	CHIP
METHOD/RANK	10
FUNCTION	IDENTITY
INSTANCE/OF	(METHOD)
METHOD/OF	(DB/PURPOSE)

891 - (get the means of transportation for a trip)

APPLICABILITY/TEST	TRIPWITHLEGS?
ARGUMENT/PATHS	(X NODE LEGS MODE/OF/TRANSPORT)
CREATOR	CHIP
METHOD/RANK	10
FUNCTION	IDENTITY
INSTANCE/OF	(METHOD)
METHOD/OF	(MODE/OF/TRANSPORT)

892 - (Get the starting point for a trip)

APPLICABILITY/TEST	TRIPWITHLEGS?
ARGUMENT/PATHS	(U NODE LEGS STARTING/POINT)
CREATOR	CHIP
METHOD/RANK	10
FUNCTION	CAR
INSTANCE/OF	(METHOD)
METHOD/OF	(STARTING/POINT)

893 - (Get the beginning time for a trip)

```

APPLICABILITY/TEST HAS-TIME?
ARGUMENT/PATHS      (U NODE TIME BEGIN/TIME)
CREATOR              CHIP
METHOD/RANK          10
FUNCTION              [LAMBDA (U)
                      (COND ((LISTP U)
                             (CAR U))
                             (T U)]

```

```

INSTANCE/OF          (METHOD)
METHOD/OF             (BEGIN/TIME)

```

394 - (Get the ending time for a trip)

```

APPLICABILITY/TEST HAS-TIME?
ARGUMENT/PATHS      (U NODE TIME END/TIME)
CREATOR              CHIP
METHOD/RANK          10
FUNCTION              [LAMBDA (U)
                      (COND ((LISTP U)
                             (CAR (LAST U)))
                             (T U)]

```

```

INSTANCE/OF          (METHOD)
METHOD/OF             (END/TIME)

```

895 - (Get the time of a trip)

```

APPLICABILITY/TEST [LAMBDA (NODE LINK VALUE)
                    (INSTANCE/OF? NODE (QUOTE DB/TRIP)]
FUNCTION            [LAMBDA (X Y)
                    (COND ((AND X Y)
                            (APPLY* (FUNCTION BUILD:)
                                     (QUOTE TIME/PERIOD)
                                     (LIST (QUOTE TIME/OF)
                                           NODE))
                            (ADD: LASTTIME/PERIOD
                                   (QUOTE BEGIN/TIME)
                                   X)
                            (ADD: LASTTIME/PERIOD
                                   (QUOTE END/TIME)
                                   Y)]

```

```

ARGUMENT/PATHS      (X NODE BEGIN/TIME)
                    (Y NODE END/TIME)
CREATOR              CHIP
METHOD/RANK          30
INSTANCE/OF          (METHOD)
METHOD/OF             (TIME)

```

896 - (Get a round trip fare)

```

FUNCTION              [LAMBDA (X1)
                      (ITIMES X1 2)]
ARGUMENT/PATHS        (X1 NODE FARE)
CREATOR               CHIP
METHOD/RANK            30
INSTANCE/OF            (METHOD)
METHOD/OF              (ROUND/TRIP/FARE)

```

897 - (Get the number of days)

```

ARGUMENT/PATHS        (X NODE DURATION DAYS)
CREATOR               CHIP
FUNCTION              IDENTITY
METHOD/RANK            30
INSTANCE/OF            (METHOD)
METHOD/OF              (DAYS)

```

956 - (Follow back pointers from time point to db/conference)

```
ARGUMENT/PATHS      (X NODE BEGIN/TIME/OF TIME/OF)
APPLICABILITY/TEST  [LAMBDA (NODE LINK VALUE)
                     (INSTANCE/OF? NODE (QUOTE TIME/POINT])

CREATOR             CHIP
METHOD/RANK         30
FUNCTION            [LAMBDA (X)
                     (CAR (SOME X (FUNCTION
                                   (LAMBDA (Z)
                                     (INSTANCE/OF?
                                       Z
                                       (QUOTE
                                         DB/CONFERENCE])

INSTANCE/OF         (METHOD)
METHOD/OF           (DB/CONFERENCE)
```

1017 - (#TRAVELERS IS 1 IF THE TRAVELER IS SPECIFIED)

```
APPLICABILITY/TEST [LAMBDA (NODE/LINK VALUE)
                     (INSTANCE/OF? NODE (QUOTE DB/TRIP])
ARGUMENT/PATHS     (X NODE TRAVELER)
FUNCTION           [LAMBDA (X)
                     (COND (X 1)
                           (T 0])

CREATOR            LAURA
METHOD/RANK        30

INSTANCE/OF        (METHOD)
METHOD/OF          (#TRAVELERS)
```

1025 - (Follow back pointers from time point to db/trip)

```
APPLICABILITY/TEST [LAMBDA (NODE LINK VALUE)
                     (INSTANCE/OF? NODE (QUOTE TIME/POINT])
ARGUMENT/PATHS     (X NODE BEGIN/TIME/OF TIME/OF)
FUNCTION           TRIPOF
CREATOR            LAURA
METHOD/RANK        30

INSTANCE/OF        (METHOD)
METHOD/OF          (DB/TRIP)
```

1047 - (Return number of travelers if traveler not specified)

```
ARGUMENT/PATHS     (X NODE TRAVELERS)
APPLICABILITY/TEST [LAMBDA (NODE LINK VALUE)
                     (INSTANCE/OF? NODE (QUOTE DB/TRIP])

CREATOR            CHIP
FUNCTION           IDENTITY
METHOD/RANK        20

INSTANCE/OF        (METHOD)
METHOD/OF          (TRAVELER)
```

1049 - (The per diem of a trip or leg of trip is the per diem of its destination)

```
APPLICABILITY/TEST HAS-DESTINATION?
ARGUMENT/PATHS     (X NODE DESTINATION PER/DIEM)
CREATOR            CHIP
FUNCTION           IDENTITY
METHOD/RANK        20

INSTANCE/OF        (METHOD)
METHOD/OF          (PER/DIEM)
```

1050 - (Finds the MEMBERS list for the CITY/PAIR corresponding to a fare.)

APPLICABILITY/TEST [LAMBDA (NODE LINK VALUE)
(INSTANCE/OF? NODE (QUOTE DB/FARE])
ARGUMENT/PATHS (X NODE FARE/OF MEMBERS)
CREATOR CHIP
FUNCTION IDENTITY
METHOD/RANK 40

INSTANCE/OF (METHOD)
METHOD/OF (DESTINATION) (STARTING/POINT)

1052 - (Get the fare for a trip)

FUNCTION GETFARES
ARGUMENT/PATHS (S NODE STARTING/POINT)
(DLIST NODE DESTINATION)
(MLIST NODE MODE/OF/TRANSPORT)
APPLICABILITY/TEST [LAMBDA (NODE LINK VALUE)
(INSTANCE/OF? NODE (QUOTE DB/TRIP])
CREATOR CHIP
METHOD/RANK 25

INSTANCE/OF (METHOD)
METHOD/OF (FARE)

1056 - (Daily expenses for a trip)

ARGUMENT/PATHS (X NODE PER/DIEM)
(Y NODE DAYS)
APPLICABILITY/TEST [LAMBDA (NODE LINK VALUE)
(INSTANCE/OF? NODE (QUOTE DB/TRIP])
FUNCTION [LAMBDA (X Y)
(ITIMES X Y)
CREATOR CHIP
METHOD/RANK 25

INSTANCE/OF (METHOD)
METHOD/OF (EXPENSES)

1058 - (The cost of a trip)

ARGUMENT/PATHS (X NODE FARE)
(Y NODE EXPENSES)
(Z NODE TRAVELERS)
APPLICABILITY/TEST [LAMBDA (NODE LINK VALUE)
(INSTANCE/OF? NODE (QUOTE (TRAVEL/PLAN
DB/TRIP])
FUNCTION [LAMBDA (X Y Z)
(PROG ((SPEAKER (SREF (QUOTE TBMA]
(RETURN
(APPLY# (FUNCTION BUILD:)
(QUOTE DB/COST)
(LIST (QUOTE VALUE)
(ITIMES Z
(IPLUS X Y)))
(LIST (QUOTE MODALITY)
(SREF (QUOTE ESTIMATED)))
(LIST (QUOTE COST/OF)
NODE)
(LIST (QUOTE UNITS)
(SREF (QUOTE DOLLARS])

CREATOR CHIP
METHOD/RANK 25

INSTANCE/OF (METHOD)
METHOD/OF (COST)

1178 - (Get the UNITS for a node with VALUE link)

```
FUNCTION          [LAMBDA (NODE)
                  (COND ((INSTANCE/OF? NODE
                                (QUOTE (DB/COST
                                         ALLOCATION
                                         DB/FEE
                                         DB/CONTRACT
                                         DB/BUDGET
                                         BUDGET/ITEM)))
                          (SREF (QUOTE DOLLARS)))
                        ((INSTANCE/OF? NODE (QUOTE
                                              CITY/PAIR))
                          (SREF (QUOTE MILES))

CREATOR           CHIP
METHOD/RANK       20

METHOD/OF         (UNITS)
INSTANCE/OF       (METHOD)
```

1189 - (Get the project associated with a budget)

```
ARGUMENT/PATHS   (X NODE BUDGET/OF PROJECT)
FUNCTION          IDENTITY
CREATOR          CHIP
METHOD/RANK       20

INSTANCE/OF       (METHOD)
METHOD/OF         (PROJECT)
```

1190 - (Get the sponsor associated with a budget)

```
ARGUMENT/PATHS   (X NODE BUDGET/OF SPONSOR)
FUNCTION          IDENTITY
CREATOR          CHIP
METHOD/RANK       20

METHOD/OF         (SPONSOR)
INSTANCE/OF       (METHOD)
```

1192 - (Get the budget for a trip)

```
FUNCTION          [LAMBDA
                  (NODE)
                  (PROG
                    [(BUDGET
                      (COND ((EQ (GET: NODE (SREF (QUOTE
                                                    MODALITY)))
                              NIL T)
                            (QUOTE TAKEN))
                          581)
                     (T 981)]
                    (ADD: NODE (SREF (QUOTE ITEM/OF))
                          BUDGET)
                    (RETURN BUDGET])

CREATOR           CHIP
METHOD/RANK       10

METHOD/OF         (ITEM/OF)
INSTANCE/OF       (METHOD)
```

1193 - (BEGIN/TIME of trip is BEGIN/TIME of first leg)

```
ARGUMENT/PATHS   (U NODE LEGS TIME BEGIN/TIME)
CREATOR          CHIP
METHOD/RANK       5
FUNCTION          CAR
APPLICABILITY/TEST TRIPWITHLEGS?

METHOD/OF         (BEGIN/TIME)
INSTANCE/OF       (METHOD)
```

1194 - (END/TIME of trip is END/TIME of last leg)

ARGUMENT/PATHS (U NODE LEGS TIME END/TIME)
CREATOR CHIP
METHOD/RANK 5
FUNCTION [LAMPDA (U)
(CAR (LAST U)
APPLICABILITY/TEST TRIPWITHLEGS?

METHOD/OF (END/TIME)
INSTANCE/OF (METHOD)

1312 - (Obsolete method left from earlier representation of fares)

ARGUMENT/PATHS (U NODE END/POINTS)
APPLICABILITY/TEST DB/FARE?
CREATOR CHIP
FUNCTION CAR
METHOD/RANK 25

INSTANCE/OF (METHOD)
METHOD/OF (FARE/START)

1313 - (Obsolete method left from earlier representation of fares)

ARGUMENT/PATHS (X NODE END/POINTS)
APPLICABILITY/TEST DB/FARE?
CREATOR CHIP
FUNCTION CADR
METHOD/RANK 25

INSTANCE/OF (METHOD)
METHOD/OF (FARE/END)

1314 - (Get money remaining from the allocation node)

APPLICABILITY/TEST HAS-ALLOCATION?
ARGUMENT/PATHS (X NODE MONEY/ALLOCATED MONEY/REMAINING)
CREATOR CHIP
FUNCTION IDENTITY

INSTANCE/OF (METHOD)
METHOD/OF (MONEY/REMAINING)

1321 - (Get current allocation from the allocation node)

ARGUMENT/PATHS (X NODE MONEY/ALLOCATED CURRENT/ALLOCATION)
CREATOR CHIP
APPLICABILITY/TEST HAS-ALLOCATION?
FUNCTION IDENTITY

INSTANCE/OF (METHOD)
METHOD/OF (CURRENT/ALLOCATION)

1322 - (Get initial allocation from the allocation node)

ARGUMENT/PATHS (X NODE MONEY/ALLOCATED INITIAL/ALLOCATION)
CREATOR CHIP
APPLICABILITY/TEST HAS-ALLOCATION?
FUNCTION IDENTITY

INSTANCE/OF (METHOD)
METHOD/OF (INITIAL/ALLOCATION)

1327 - (Get money spent from the allocation node)

ARGUMENT/PATHS (X NODE MONEY/ALLOCATED MONEY/SPENT)
CREATOR CHIP
FUNCTION IDENTITY
APPLICABILITY/TEST HAS-ALLOCATION?

INSTANCE/OF (METHOD)
METHOD/OF (MONEY/SPENT)

1336 - (Expenses for a leg of a trip are per/diem * #days)

```

FUNCTION          PERDAYEXPENSES
ARGUMENT/PATHS    {X NODE PER/DIEM)
                  {Y NODE #DAYS)
                  {Z NODE DESTINATION)
                  {W NODE LEG/OF TRAVELER)
APPLICABILITY/TEST [LAMBDA (NODE)
                  (INSTANCE/OF? NODE (QUOTE LEG/OF/TRIP])
CREATOR           CHIP
INSTANCE/OF       (METHOD)
METHOD/OF         (EXPENSES)

```

1337 - (The expenses for a trip are the sum of the expenses of the legs)

```

APPLICABILITY/TEST [LAMBDA (NODE)
                  (INSTANCE/OF? NODE (QUOTE DB/TRIP])
ARGUMENT/PATHS    (U NODE LEGS EXPENSES)
CREATOR           CHIP
METHOD/RANK       20
FUNCTION          [LAMBDA (U)
                  (APPLY (FUNCTION IPLUS)
                           U)]

```

```

INSTANCE/OF       (METHOD)
METHOD/OF         (EXPENSES)

```

1340 - (The fare for a leg of a trip)

```

ARGUMENT/PATHS    {S NODE STARTING/POINT)
                  {D NODE DESTINATION)
                  {M NODE MODE/OF/TRANSPORT)
APPLICABILITY/TEST [LAMBDA (NODE)
                  (INSTANCE/OF? NODE (QUOTE LEG/OF/TRIP])
CREATOR           CHIP
FUNCTION          GETFARES1
INSTANCE/OF       (METHOD)
METHOD/OF         (FARE)

```

1341 - (The cost of a trip is fare plus expenses)

```

FUNCTION          [LAMBDA (X Y)
                  (IPLUS X Y)]
ARGUMENT/PATHS    {X NODE FARE)
                  {Y NODE EXPENSES)
APPLICABILITY/TEST [LAMBDA (NODE)
                  (INSTANCE/OF? NODE (QUOTE LEG/OF/TRIP])
CREATOR           CHIP
INSTANCE/OF       (METHOD)
METHOD/OF         (COST)

```

1342 - (The cost of a trip is the sum of the costs of the legs)

```
ARGUMENT/PATHS (X NODE LEGS COST)
FUNCTION [LAMBDA (X)
  (PROG
    ((SPEAKER (SREF (QUOTE TBMA)
      (RETURN
        (APPLY*
          (FUNCTION BUILD:)
          (QUOTE DB/COST)
          (LIST (QUOTE VALUE)
            (APPLY (FUNCTION IPLUS) X))
          (LIST (QUOTE MODALITY)
            (SREF (QUOTE ESTIMATED)))
          (LIST (QUOTE COST/OF)
            (NODE)
            (LIST (QUOTE UNITS)
              (SREF (QUOTE DOLLARS))
              (INSTANCE/OF? NODE (QUOTE DB/TRIP))
            )
          )
        )
      )
    )
  )
)
APPLICABILITY/TEST [LAMBDA (NODE)
CREATOR CHIP
METHOD/RANK 10
INSTANCE/OF (METHOD)
METHOD/OF (COST)
```

1349 - (A person's location is the location of his/her group)

```
ARGUMENT/PATHS (X NODE MEMBER/OF)
APPLICABILITY/TEST [LAMBDA (NODE)
  (INSTANCE/OF? NODE (QUOTE PERSON))
CREATOR CHIP
INSTANCE/OF (METHOD)
METHOD/OF (LOCATION)
```

1350 - (Find a budget item to cover a trip)

```
ARGUMENT/PATHS (TRIP NODE)
  (BUDGET NODE TRAVELER MEMBER/OF GROUP/OF BUDGET)
CREATOR CHIP
FUNCTION FIND-HOME
INSTANCE/OF (METHOD)
METHOD/OF (COVERED/BY)
```

1396 - ((GET: item 'SELF) => item)

```
ARGUMENT/PATHS (X NODE)
CREATOR CHIP
FUNCTION IDENTITY
INSTANCE/OF (METHOD)
METHOD/OF (SELF)
```

1397 - (The name of an item)

```
ARGUMENT/PATHS (X NODE INSTANCE/OF ENGLISH/NAME PNAME)
CREATOR CHIP
INSTANCE/OF (METHOD)
METHOD/OF (TYPENAME)
```

2449 - (GET THE INSTANCES OF THE KINDS)

```
ARGUMENT/PATHS (L NODE KINDS INSTANCES)
CREATOR CHIP
FUNCTION [LAMBDA (L)
  (APPLY (FUNCTION APPEND) L)
INSTANCE/OF (METHOD)
METHOD/OF (INSTANCES)
```

2460 - (Return the stative verb)

FUNCTION	[LAMBDA {X}
	{SREF (QUOTE IS]
ARGUMENT/PATHS	{X NODE}
CREATOR	CHIP
METHOD/RANK	30
INSTANCE/OF	{METHOD}
METHOD/OF	{ACTION}

2461 - (Return the ACTION of the type node)

APPLICABILITY/TEST	[LAMBDA {X}
	{NOT (PAST? X]
FUNCTION	IDENTITY
ARGUMENT/PATHS	{X NODE INSTANCE/OF ACTION}
CREATOR	CHIP
METHOD/RANK	10
INSTANCE/OF	{METHOD}
METHOD/OF	{ACTION}

2475 - (Return the past form of the verb for the action)

APPLICABILITY/TEST	[LAMBDA {X}
	{PAST? X]
ARGUMENT/PATHS	{X NODE INSTANCE/OF ACTION PAST/FORM}
CREATOR	CHIP
FUNCTION	IDENTITY
METHOD/RANK	20
INSTANCE/OF	{METHOD}
METHOD/OF	{ACTION}
LAST/MENTIONED/OF	{METHOD}

1.4 Generation Frames

This section contains the complete set of generation frames used in response generation (section F). They are roughly grouped by the data base items to which they apply, e.g. ALLOCATION, BUDGET/ITEM, DB/TRIP, CITY, DB/BUDGET, DB/CONFERENCE, etc.

1402

FRAME	(EXP "There") (V "are") (ADJ CURRENT/ALLOCATION) (N UNITS) (V "allocated") (PP ALLOCATION/OF *OPT*) (CONJ "and") (ADJ MONEY/REMAINING) (N UNITS) (ADJ "remaining")
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(ALLOCATION)

1403

FRAME	(DET "the") (N TYPENAME) (PP PLANS)
CREATOR	CHIP
TYPE	NP
PRECONDITION	(MEDIUM?)
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(BUDGET/ITEM)

1404

PRECONDITION	(OR (SHORT?) (MEDIUM?))
FRAME	(N TYPENAME) (N (FUNCTION [LAMBDA (X) (LIST (QUOTE N) X]) SELF))
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(BUDGET/ITEM)

1405

PRECONDITION	(LINKFOL ITEM (QUOTE PLANS))
FRAME	(NP SELF (MODE SHORT?)) (V ACTION) (PP PLANS)
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(BUDGET/ITEM)

1406

PRECONDITION	(NULL (LINKFOL ITEM (QUOTE PLANS)))
FRAME	(NP SELF (MODE SHORT))
	(V ACTION)
	(PREP "for")
	(ADJ "miscellaneous")
	(N "charges")
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(BUDGET/ITEM)

1407

FRAME	(S SELF)
	(S MONEY/ALLOCATED (OMIT ALLOCATION/OF))
CREATOR	CHIP
TYPE	SS
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(BUDGET/ITEM)

1408

FRAME	(PREP (DEFAULT "in"))
	(NP SELF)
CREATOR	CHIP
TYPE	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(BUDGET/ITEM)

1409

FRAME	(PREP (DEFAULT "in"))
	(NP SELF)
CREATOR	CHIP
TYPE	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(CITY)

1410

FRAME	(DET "The")
	(ADJ "travel")
	(N TYPENAME)
	(PP BUDGET/OF (USE "for"))
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/BUDGET)

1411

FRAME	(NP SELF)
	(V ACTION)
	(ADJ MONEY/REMAINING)
	(N UNITS)
	(ADJ "left")
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/BUDGET)

1412

FRAME	(S SELF)
CREATOR	(SS (FUNCTION DBUDGET/ITEMS ITEMS))
TYPE	CHIP
	SS
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/BUDGET)

1413

FRAME	(NP SELF)
	(V ACTION)
	(PP LOCATION *OPT*)
	(PP TIME (OMIT TIME/OF)
	OPT)
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/CONFERENCE)

1414

FRAME	(DET "the")
	(ADJ TIME *OPT*)
	(ADJ SPONSOR)
	(N TYPENAME)
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/CONFERENCE)

1415

FRAME	(PREP (DEFAULT "at"))
	(NP SELF)
CREATOR	CHIP
TYPE	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/CONFERENCE)

1416

FRAME	(NP SELF)
	(V ACTION)
	(ADJ TRAVEL/MONEY)
	(N UNITS)
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/CONTRACT)

1417

FRAME	(DET "The")
	(ADJ YEAR)
	(ADJ SPONSOR)
	(ADJ PROJECT)
	(N TYPENAME)
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/CONTRACT)

1418

FRAME	(PREP (DEFAULT "for"))
CREATOR	(NP SELF)
TYPE	CHIP
	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/CONTRACT)

1419

FRAME	(DET "The")
	(ADJ MODALITY)
	(N TYPENAME)
	(PP COST/OF (USE "of"))
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/COST)

1420

FRAME	(NP SELF)
	(V "is")
	(ADJ VALUE)
	(N UNITS)
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/COST)

1421

FRAME	(DET "The")
	(ADJ TYPE)
	(ADJ MODE/OF/TRANSPORT)
	(N TYPENAME)
	(PP FARE/START (USE "from"))
	(PP FARE/END (USE "to"))
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/FARE)

1422

FRAME	(NP SELF)
	(V ACTION)
	(ADJ VALUE)
	(N UNITS)
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/FARE)

1423

FRAME	(DET "The")
	(ADJ "registration")
	(N TYPENAME)
	(PP REGISTRATION/FEE/OF (USE "for"))
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(DB/FEE)

1424

FRAME	(NP SELF) (V ACTION) (ADJ VALUE) (N UNITS)
CREATOR TYPE	CHIP S
INSTANCE/OF G-FRAME/OF	(G-FRAME) (DB/FEE)

1425

FRAME	(NUMBER TRIP *OPT*) (NP TRAVELER *OPT*) (PREP "to") (NP (FUNCTION GENLIST DESTINATION)) (PP TIME *OPT*)
CREATOR TYPE PRECONDITION	CHIP S (SHORT?)
INSTANCE/OF G-FRAME/OF	(G-FRAME) (DB/TRIP)

1426

FRAME	(NP TRAVELER) (V ACTION) (BP SELF)
CREATOR TYPE	CHIP S
INSTANCE/OF G-FRAME/OF	(G-FRAME) (DB/TRIP)

1427

FRAME	(PREP (DEFAULT "of")) (NP SELF)
CREATOR TYPE	CHIP PP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (DB/TRIP)

1428

FRAME	(POSS TRAVELER) (N TYPENAME) (BP SELF)
CREATOR TYPE	CHIP NP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (DB/TRIP)

1429

FRAME	(POSS TRAVELER) (N TYPENAME)
CREATOR TYPE PRECONDITION	CHIP NP (SHORT?)
INSTANCE/OF G-FRAME/OF	(G-FRAME) (DB/TRIP)

1430

FRAME	(PP STARTING/POINT (USE "from")) (PREP "to") (NPL (FUNCTION GENLIST DESTINATION)) (PP TIME *OPT*)
CREATOR TYPE	CHIP BP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (DB/TRIP)

1431

FRAME	(PREP (DEFAULT "by")) (NP SELF)
CREATOR TYPE	CHIP PP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (FARE/TYPE)

1432

FRAME CREATOR TYPE	(N (FUNCTION DLENGTH/OF/TIME)) CHIP NP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (LENGTH/OF/TIME)

1433

FRAME CREATOR TYPE	(PREP (DEFAULT "for")) (NP SELF) CHIP PP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (LENGTH/OF/TIME)

1434

FRAME	(NP (FUNCTION [LAMBDA (X) (DNAME X T] SELF))
CREATOR TYPE	CHIP POSS
INSTANCE/OF G-FRAME/OF	(G-FRAME) (PERSON)

1435

FRAME	(QWORD "What") (V "is") (DET "the") (N ARG1) (PREP "for") (DET "this") (N ARG2)
CREATOR TYPE	CHIP S
INSTANCE/OF G-FRAME/OF	(G-FRAME) (Q1)

1436

FRAME (QWORD "When")
 { V "is"
 { DET "the"
 { N "traveler"
 { V "leaving"
 { NP ARG1)
 CREATOR CHIP
 TYPE S
 INSTANCE/OF (G-FRAME)
 G-FRAME/OF (Q2)

1437

FRAME (QWORD "When")
 { V "is"
 { DET "the"
 { N "traveler"
 { V "arriving"
 { PP ARG1 (USE "in"))
 CREATOR CHIP
 TYPE S
 INSTANCE/OF (G-FRAME)
 G-FRAME/OF (Q3)

1438

FRAME (NP "What")
 { V "is"
 { DET "the"
 { N "fare"
 { PP ARG1 (USE "from"))
 { PP ARG2 (USE "to"))
 { PP ARG3 (USE "by"))
 CREATOR CHIP
 TYPE S
 INSTANCE/OF (G-FRAME)
 G-FRAME/OF (Q4)

1439

FRAME (AUX "Do")
 { NP "you"
 { V "mean"
 { NPL (FUNCTION [LAMBDA (X)
 { GENLIST X (QUOTE OR)
 ARG4))
 CREATOR CHIP
 TYPE S
 INSTANCE/OF (G-FRAME)
 G-FRAME/OF (Q5)

1440

FRAME (NP "Whom")
 { AUX "do"
 { NP "you"
 { V "mean"
 CREATOR CHIP
 TYPE S
 INSTANCE/OF (G-FRAME)
 G-FRAME/OF (Q6)

1441

FRAME	(ADJ "What") (N ARG1) (AUX "should") (N "I") (V "use") (PREP "in") (DET "the") (ADJ ARG2) (N "slot")
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(Q7)

1442

FRAME	(ADJ "Which") (N "value") (AUX "should") (N "I") (V "use") (PUNC ":") (N "OLD") (PUNC "=") (NP ARG1) (PUNC " ") (N "NEW") (PUNC "=") (NP ARG2) (CONJ "or") (N "BOTH")
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(Q8)

1443

FRAME	(AUX "Is") (NP ARG1) (V "covered") (PP ARG2 (USE "by"))
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(Q9)

1444

FRAME	(DET "The") (ADJ "following") (N ARG1) (N "description") (AUX "has") (AUX "been") (V "completed")
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(S1)

1445

FRAME	(PRON "I") (AUX "cannot") (V "find") (DET "a") (N "referent") (PREP "for") (NP (FUNCTION GPRIN1 ARG1))
CREATOR TYPE	CHIP S
INSTANCE/OF G-FRAME/OF	(G-FRAME) (S2)

1446

FRAME	(NP "I") (VP "have") (DET "a") (ADJ "previous") (N "value") (PREP "of") (NP ARG1) (ADJ "stored")
CREATOR TYPE	CHIP S
INSTANCE/OF G-FRAME/OF	(G-FRAME) (S3)

1447

FRAME	(NP TIME/OF (OMIT TIME)) (V "is") (PP SELF)
CREATOR TYPE	CHIP S
INSTANCE/OF G-FRAME/OF	(G-FRAME) (TIME/PERIOD)

1448

PRECONDITION FRAME	(NOT (POINT-TIME? ITEM)) (PP BEGIN/TIME (USE "from")) (PPY END/TIME (USE "to"))
CREATOR TYPE	CHIP PP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (TIME/PERIOD)

1449

FRAME CREATOR TYPE	(PP BEGIN/TIME) CHIP PP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (TIME/PERIOD)

1450

FRAME CREATOR TYPE	(PP END/TIME) CHIP PP
INSTANCE/OF G-FRAME/OF	(G-FRAME) (TIME/PERIOD)

1451

FRAME	(PP DURATION (USE "for"))
CREATOR	CHIP
TYPE	PP

INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/PERIOD)

1452

FRAME	(ADJ BEGIN/TIME)
CREATOR	CHIP
TYPE	ADJ

INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/PERIOD)

1453

PRECONDITION	(HAS-DAYS? ITEM)
FRAME	(PREP (DEFAULT "on"))
	(NPY SELF)
CREATOR	CHIP
TYPE	PP

INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/POINT)

1454

FRAME	(NP SELF)
CREATOR	CHIP
TYPE	PP

INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/POINT)

1455

FRAME	(NP (FUNCTION DTIME/POINT SELF))
CREATOR	CHIP
TYPE	NP

INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/POINT)

1456

FRAME	(PREP (DEFAULT "on"))
	(NPY SELF)
CREATOR	CHIP
TYPE	PPY

INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/POINT)

1457

PRECONDITION	(NEWYEAR ITEM)
FRAME	(NP SELF)
	(PUNC ",")
	(NUMBER YEAR)
CREATOR	CHIP
TYPE	NPY

INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/POINT)

1458

FRAME	(NP SELF)
CREATOR	CHIP
TYPE	NPY
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/POINT)

1459

FRAME	(N (FUNCTION DMONTH SELF))
	(N YEAR)
CREATOR	CHIP
TYPE	ADJ
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TIME/POINT)

1460

FRAME	(NUMBER TRIP *OPT*)
	{ N TRAVELERS *OPT* }
	(PREP "to")
	(NP (FUNCTION GENLIST DESTINATION))
	(PP TIME *OPT*)
CREATOR	CHIP
TYPE	S
PRECONDITION	(SHORT?)
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TRAVEL/PLAN)

1461

PRECONDITION	(IGREATERP (LINKFOL ITEM (QUOTE TRAVELERS))
	1)
FRAME	(ADJ #TRAVELERS)
	(N "people")
	(V ACTION)
	(BP SELF)
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TRAVEL/PLAN)

1462

FRAME	(ADJ TRAVELERS)
	(N "person")
	(V "is going")
	(EP SELF)
CREATOR	CHIP
TYPE	S
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TRAVEL/PLAN)

1463

FRAME	(N #TRAVELERS)
	(BP SELF)
CREATOR	CHIP
TYPE	NP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TRAVEL/PLAN)

1464

FRAME	(PREP "for")
CREATOR	(NP SELF)
TYPE	CHIP
	PP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TRAVEL/PLAN)

1465

FRAME	(PREP "to")
	(NPL (FUNCTION GENLIST DESTINATION))
CREATOR	(PP TIME *OPT*)
TYPE	CHIP
	BP
INSTANCE/OF	(G-FRAME)
G-FRAME/OF	(TRAVEL/PLAN)

1.5 A Generated Description of the Stored Trips

TRAVELNET contains a fairly complete representation of the BBN speech understanding group's travel budget for the period from November, 1974 to December, 1975. This section contains a set of descriptions of the trips taken during that period. The descriptions were produced by the response generation program. Note that years are given only when the year changes from one date description to another. The program does this to avoid needless repetition in a dialogue setting. The descriptions were produced by invoking the following command:

```
(for: every x / (find: (instance/of 'db/trip)
                      (modality 'taken)) : t ;
      (output: x]
```

JERRY WOLF WENT from BOSTON, MASSACHUSETTS to WASHINGTON D.C. on May 1st, 1975.

JERRY WOLF WENT from BOSTON, MASSACHUSETTS to ST LOUIS, MISSOURI from November 5th, 1974 to November 8th.

BILL WOODS WENT from BOSTON, MASSACHUSETTS to SAN DIEGO, CALIFORNIA from March 11th, 1975 to March 16th.

BILL WOODS WENT from BOSTON, MASSACHUSETTS to LOS ANGELES from January 25th to February 1st.

BILL WOODS WENT from BOSTON, MASSACHUSETTS to SAN FRANCISCO, CALIFORNIA from January 13th to January 17th.

BILL WOODS WENT from BOSTON, MASSACHUSETTS to WASHINGTON D.C. from January 7th to January 8th.

BILL WOODS WENT from BOSTON, MASSACHUSETTS to NEW YORK CITY, NEW YORK on December 6th, 1974.

BILL WOODS WENT from BOSTON, MASSACHUSETTS to WASHINGTON D.C. on December 3rd.

BILL WOODS WENT from BOSTON, MASSACHUSETTS to WHITE PLAINS, NEW YORK on November 22nd.

JOHN MAKHOUL WENT from BOSTON, MASSACHUSETTS to ST LOUIS, MISSOURI from November 5th to November 8th.

JOHN MAKHOUL WENT from BOSTON, MASSACHUSETTS to LOS ANGELES, and SANTA BARBARA, CALIFORNIA from December 8th to December 10th.

JOHN MAKHOUL WENT from BOSTON, MASSACHUSETTS to WASHINGTON D.C. from February 2nd, 1975 to February 4th.

JOHN MAKHOUL WENT from BOSTON, MASSACHUSETTS to AUSTIN, TEXAS from April 6th to April 11th.

JOHN MAKHOUL WENT from BOSTON, MASSACHUSETTS to PARIS, FRANCE, GRENOBLE, FRANCE, and TURIN, ITALY from June 9th to June 14th.

LYNN COSELL WENT from BOSTON, MASSACHUSETTS to SAN FRANCISCO, CALIFORNIA from July 27th to July 30th.

LYNN COSELL WENT from BOSTON, MASSACHUSETTS to LOS ANGELES, and SANTA BARBARA, CALIFORNIA from December 8th, 1974 to December 13th.

JOHN MAKHOUL WENT from BOSTON, MASSACHUSETTS to PHILADELPHIA, PENNSYLVANIA from February 24th, 1975 to February 25th.

BILL MERRIAM WENT from BOSTON, MASSACHUSETTS to WASHINGTON D.C., LOS ANGELES, and SAN FRANCISCO CALIFORNIA from March 14th to March 22nd.

JOE BECKER WENT from BOSTON, MASSACHUSETTS to SAN FRANCISCO, CALIFORNIA from March 17th to March 23rd.

VICTOR ZUE WENT from BOSTON, MASSACHUSETTS to SANTA BARBARA, CALIFORNIA from February 18th to February 23rd.

JOHN MAKHOUL WENT from BOSTON, MASSACHUSETTS to WASHINGTON D.C. on October 16th.

BILL WOODS WENT from BOSTON, MASSACHUSETTS to JUAREZ, MEXICO from October 15th to October 17th.

JOHN MAKHOUL WENT from BOSTON, MASSACHUSETTS to BAGHDAD, IRAQ, and BEIRUT, LEBANON from November 25th to December 11th.

JERRY WOLF WENT from BOSTON, MASSACHUSETTS to SAN FRANCISCO, CALIFORNIA from November 3rd to November 8th.

RICH SCHWARTZ WENT from BOSTON MASSACHUSETTS to SAN FRANCISCO CALIFORNIA from November 3rd to November 8th.

Official Distribution List

Contract N00014-75-C-0533

Defense Documentation Center
Cameron Station
Alexandria, Virginia 22314

Office of Naval Research
Information Systems Program
Code 437
Arlington, Virginia 22217

Office of Naval Research
Code 1021P
Arlington, Virginia 22217

Office of Naval Research
Branch Office, Boston
495 Summer Street
Boston, Massachusetts 02210

Office of Naval Research
Branch Office, Chicago
536 South Clark Street
Chicago, Illinois 60605

Office of Naval Research
Branch Office, Pasadena
1030 East Green Street
Pasadena, California 91106

New York Area Office
715 Broadway - 5th Floor
New York, New York 10003

Naval Research Laboratory
Technical Information Division
Code 2627
Washington, D.C. 20375

Dr. A. L. Slafkosky
Scientific Advisor
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

Office of Naval Research
Code 455
Arlington, Virginia 22217

Office of Naval Research
Code 458
Arlington, Virginia 22217

Naval Electronics Lab. Center
Advanced Software Technology Division
Code 5200
San Diego, California 92152

Mr. E. H. Gleissner
Naval Ship Research and
Development Center
Computation and Mathematics Dept.
Bethesda, Maryland 20084

Captain Grace M. Hopper
NAICOM/MIS Planning Branch (OP-916D)
Office of Chief of Naval Operations
Washington, D.C. 20350

Mr. Kin B. Thompson
Technical Director
Information Systems Division (OP-91T)
Office of Chief of Naval Operations
Washington, D.C. 20350

Advanced Research Projects Agency
Information Processing Techniques
1400 Wilson Boulevard
Arlington, Virginia 22209

Commanding Officer
Naval Air Development Center
Warminster, Pennsylvania 18974

Professor Omar Wing
Dept. of Electrical Engineering
Columbia University
New York, New York 10027

Assistant Chief for Technology
Office of Naval Research
Code 200
Arlington, Virginia 22217